

# Computación y Sistemas

Vol. 23 No. 3 July – September, 2019

<b>Computational Linguistics: Introduction to the Thematic Issue</b>	<b>625</b>
<i>Alexander Gelbukh</i>	

## **Thematic Issue: Computational Linguistics**

<b>Ontological Knowledge for Rhetorical Move Analysis</b>	<b>633</b>
<i>Mohammed Alliheedi, Robert E. Mercer, Sandor Haas Neil</i>	

<b>Central Embeddings for Extractive Summarization Based on Similarity</b>	<b>649</b>
<i>Sandra J. Gutiérrez Hinojosa, Hiram Calvo, Marco A. Moreno Armendáriz</i>	

<b>Predicting and Integrating Expected Answer Types into a Simple Recurrent Neural Network Model for Answer Sentence Selection</b>	<b>665</b>
<i>Sanjay Kamath, Brigitte Grau, Yue Ma</i>	

<b>A Fuzzy Approach to Mute Sensitive Information in Noisy Audio Conversations</b>	<b>675</b>
<i>Imran Sheikh, Balamallikarjuna Garlapati, Srinivas Chalamala, Sunil Kumar Kopparapu</i>	

<b>A Comparative Study on Text Representation Models for Topic Detection in Arabic</b>	<b>683</b>
<i>Rim Koulali, Abdelouafi Meziane</i>	

<b>A Deep Attention based Framework for Image Caption Generation in Hindi Language</b>	<b>693</b>
<i>Rijul Dhir, Santosh Kumar Mishra, Sriparna Saha, Pushpak Bhattacharyya</i>	

<b>An Ensemble of Automatic Keyword Extractors: TextRank, RAKE and TAKE</b>	<b>703</b>
<i>Tayfun Pay, Stephen Lucci, James L. Cox</i>	

<b>Anaphoric Connectives and Long Distance Discourse Relations in Czech</b>	<b>711</b>
<i>Lucie Poláková, Jiří Mírovský</i>	
<b>Complex Named Entities Extraction on the Web: Application to Social Events</b>	<b>719</b>
<i>Armel Fotsoh, Annig Le Parc Lacayrelle, Christian Sallaberry</i>	
<b>An Exploratory Study of the Use of Senses, Syntax and Cross-Linguistic Information for Subjectivity Detection in Spanish</b>	<b>731</b>
<i>Rodrigo López, Daniel Peñaloza, Francisco Beingolea, Juanjose Tenorio, Marco Sobrevilla Cabezudo</i>	
<b>Concept Discovery through Information Extraction in Restaurant Domain</b>	<b>741</b>
<i>Nadeesha Pathirana, Sandaru Seneviratne, Rangika Samarawickrama, Shane Wolff, Charith Chitraranjan, Uthayasanker Thayasivam, Tharindu Ranasinghe</i>	
<b>Deep Semantic Role Labeling for Tweets using 5W1H: Who, What, When, Where, Why and How</b>	<b>751</b>
<i>K. Chakma, Amitava Das, Swapan Debbarma</i>	
<b>English Dataset For Automatic Forum Extraction</b>	<b>765</b>
<i>Jakub Sido, Miloslav Konopík, Ondřej Pražák</i>	
<b>Enriching Word Embeddings with Global Information and Testing on Highly Inflected Language</b>	<b>773</b>
<i>Lukas Svoboda, Tomas Bryhcín</i>	
<b>Semi-Automatic Knowledge Graph Construction by Relation Pattern Extraction</b>	<b>785</b>
<i>Yingju Xia, Zhongguang Zheng, Yao Meng, Jun Sun</i>	
<b>Identification of POS Tag for Khasi Language based on Hidden Markov Model POS Tagger</b>	<b>795</b>
<i>Sunita Warjri, Partha Pakray, Saralin Lyngdoh, Arnab Kumar Maji</i>	
<b>Extracting Context of Math Formulae Contained inside Scientific Documents</b>	<b>803</b>
<i>Amarnath Pathak, Ranjita Das, Partha Pakray, Alexander Gelbukh</i>	

<b>Identifying Repeated Sections within Documents</b>	<b>819</b>
<i>Girish K. Palshikar, Sachin Pawar, Rajiv Srivastava, Mahek Shah</i>	
<b>Identifying Short-Term Interests from Mobile App Adoption Pattern</b>	<b>829</b>
<i>Bharat Gaiind, Nitish Varshney, Shubham Goel, Akash Mondal</i>	
<b>Joint Learning of Named Entity Recognition and Dependency Parsing using Separate Datasets</b>	<b>841</b>
<i>Arda Akdemir, Tunga Gungor</i>	
<b>Kazakh Text Summarization using Fuzzy Logic</b>	<b>851</b>
<i>Altanbek Zulkhazhav, Zhanibek Kozhirkbayev, Zhandos Yessenbayevy, Altynbek Sharipbay</i>	
<b>KeyVector: Unsupervised Keyphrase Extraction using Weighted Topic via Semantic Relatedness</b>	<b>861</b>
<i>Alymzhan Toleu, Gulmira Tolegen, Rustam Mussabayev</i>	
<b>Mining Purchase Intent in Twitter</b>	<b>871</b>
<i>Rejwanul Haque, Mohammed Hasanuzzaman, Andy Way</i>	
<b>Multi-Head Multi-Layer Attention to Deep Language Representations for Grammatical Error Detection</b>	<b>883</b>
<i>Masahiro Kaneko, Mamoru Komachi</i>	
<b>My Word! Machine versus Human Computation Methods for Identifying and Resolving Acronyms</b>	<b>893</b>
<i>Christopher G. Harris, Padmini Srinivasan</i>	
<b>Ontology based Extractive Text Summarization: The Contribution of Instances</b>	<b>905</b>
<i>Murillo Lagranha Flores, Elder Rizzon Santos, Ricardo Azambuja Silveira</i>	
<b>Ontology-driven Text Feature Modeling for Disease Prediction using Unstructured Radiological Notes</b>	<b>915</b>
<i>Gokul S. Krishnan, Sowmya Kamath S.</i>	
<b>Promoting the Knowledge of Source Syntax in Transformer NMT</b>	<b>923</b>
<i>Thuong-Hai Pham, Dominik Macháček, Ondřej Bojar</i>	
<b>Reasoning over Arabic WordNet Relations with Neural Tensor Network</b>	<b>935</b>
<i>Mohamed Ali Batita, Rami Ayadi, Mounir Zrigui</i>	

<b>Refining Concepts by Machine Learning</b>	<b>943</b>
<i>Marek Menšík, Marie Duží, Adam Albert, Vojtěch Patschka, Miroslav Pajr</i>	
<b>Relation between Titles and Keywords in Japanese Academic Papers using Quantitative Analysis and Machine Learning</b>	<b>959</b>
<i>Masaki Murata, Natsumi Morimoto</i>	
<b>ReEmb: A Relevance-based Application Embedding for Mobile App Retrieval And Categorization</b>	<b>969</b>
<i>Shubham Krishna, Ahsaas Bajaj, Mukund Rungta, Hemant Tiwari, Vanraj Vala</i>	
<b>Script Independent Morphological Segmentation for Arabic Maghrebi Dialects: An Application to Machine Translation</b>	<b>979</b>
<i>Salima Harrat, Karima Meftouh, Kamel Smaili</i>	
<b>Sentence Generation Using Selective Text Prediction</b>	<b>991</b>
<i>Samarth Navali, Jyothirmayi Kolachalam, Vanraj Vala</i>	
<b>Sentence Similarity Techniques for Short vs Variable Length Text using Word Embeddings</b>	<b>999</b>
<i>D. Shashavali, V. Vishwjeet, Rahul Kumar, Gaurav Mathur, Nikhil Nihal, Siddhartha Mukherjee, Suresh Venkanagouda Patil</i>	
<b>Similarity Driven Unsupervised Learning for Materials Science Terminology Extraction</b>	<b>1005</b>
<i>Sapan Shah, Sarath S, Sreedhar Reddy</i>	
<b>Text Classification using Gated Fusion of N-Gram Features and Semantic Features</b>	<b>1015</b>
<i>Ajay Nagar, Anmol Bhasin, Gaurav Mathur</i>	
<b>The Role of WordNet Similarity in the Affective Analysis Pipeline</b>	<b>1021</b>
<i>Alejandra Segura Navarrete, Christian Vidal Castro, Clemente Rubio Manzano, Claudia Martínez Araneda</i>	
<b>Understanding Blogs through the Lens of Readers' Comments</b>	<b>1033</b>
<i>Abhilasha Sancheti, Natwar Modani, Gautam Choudhary, Chintha Priyadarshini, Sai Sandeep Moparthy</i>	
<b>Word Embeddings for IoT Based on Device Activity Footprints</b>	<b>1043</b>
<i>Kushal Singla, Joy Bose, Nitish Varshney</i>	



**Towards Simple but Efficient Next Utterance Ranking** 1055  
*Basma El Amel Boussaha, Nicolas Hernandez, Christine Jacquin, Emmanuel Morin*

**A New Approach for Twitter Event Summarization Based on Sentence Identification and Partial Textual Entailment** 1065  
*Dwijen Rudrapal, Amitava Das, Baby Bhattacharya*

**Recognizing Musical Entities in user Generated Content** 1079  
*Lorenzo Porcaro, Horacio Saggion*

**Cross-Domain Failures of Fake News Detection** 1089  
*Maria Janicka, Maria Pszona, Aleksander Wawer*

**Cochlear Mechanical Models used in Automatic Speech Recognition Tasks** 1099  
*José Luis Oropeza Rodríguez, Sergio Suárez Guerra*

## Regular Articles

**Modelado para trabes de sección transversal rectangular con cartelas parabólicas: Parte 2** 1115  
*Ricardo Sandoval Rivas, Arnulfo Luévanos Rojas, Sandra López Chavarría, Manuel Medina Elizondo*

**A Numerical Perspective on the Jaynes-Cummings Model Wigner Function** 1125  
*J. C. García-Melgarejo, N. Lozano-Crisóstomo, J. Jesús Escobedo-Alatorre, K. J. Sánchez-Pérez, M. Torres-Cisneros, E. S. Arroyo-Rivera, R. Guzmán-Cabrera*



## Computational Linguistics: Introduction to the Thematic Issue

This special issue of *Computación y Sistemas* presents a selection of papers on natural language processing and computational linguistics, along with two regular papers.

Natural language processing is an area of artificial intelligence and its applications devoted to analysis and generation of data streams involved in human communication using language, such as English or Spanish, typically in the form of text or speech, as well as, in multimodal setting, associated facial expressions and body language. It embraces natural language processing and computational linguistics in a wider context of their real-life applications in a wide range of disciplines.

The typical tasks of natural language processing include machine translation, text classification, text summarization, information extraction, and sentiment analysis, while typical applications include opinion mining, human-computer interaction, and information retrieval, among many other tasks.

**2965 José Luis Oropeza Rodríguez, Sergio Suárez Guerra** from México in their paper “Cochlear Mechanical Models used in Automatic Speech Recognition Tasks” ABSTRACT: In this paper we show that its possible unify two theories that we can find in the state of the art related with human hearing, one of them related with human perceptual phenomenon and the another one related with cochlear mechanic’s models linear.

**3234 Tayfun Pay, Stephen Lucci, James L. Cox** from United States in their paper “An Ensemble of Automatic Keyword Extractors: TextRank, RAKE and TAKE” ABSTRACT: We construct an ensemble method for automatic keyword extraction from single documents. We utilize three different unsupervised automatic keyword extractors in building our ensemble method. These three approaches provide candidate keywords for the ensemble method without using their respective threshold functions.

**3235 Armel Fotsoh, Annig Le Parc Lacayrelle, Christian Sallaberry** from France in their paper “Complex Named Entities Extraction on the Web: Application to Social Events” ABSTRACT: In this

paper, we focus on the extraction of social events in text from the web. We consider social events as complex Named Entities (NEs) i.e. NEs represented by a list of properties that can be simple values (text, number, etc.), “elementary” NEs and/or other complex NEs. Regarding the extraction of these complex NEs, our contribution focuses on the noisy context issue.

**3237 Abhilasha Sancheti, Natwar Modani, Gautam Choudhary, Chintia Priyadarshini, Sai Sandeep Moparthy** from India in their paper “Understanding Blogs Through the Lens of Readers’ Comments” ABSTRACT: In order to keep their audience engaged, authors need to make sure that the blogs or articles they write cater to the taste of their audience and are understood by them. With the rapid proliferation of online blogging websites, the participation of readers by expressing their opinions and reviews has also increased in the form of comments on the blogs.

**3238 Gokul S. Krishnan, Sowmya Kamath S.** from India in their paper “Ontology-driven Text Feature Modeling for Disease Prediction using Unstructured Radiological Notes” ABSTRACT Clinical Decision Support Systems (CDSSs) support medical personnel by offering aid in decision making and timely interventions in patient care. Typically such systems are built on structured Electronic Health Records (EHRs), which, unfortunately have a very low adoption rate in developing countries at present.

**3239 Altanbek Zulkhazhav, Zhanibek Kozhimbayev, Zhandos Yessenbayevy, Altynbek Sharipbay** in their paper “Kazakh Text Summarization using Fuzzy Logic” ABSTRACT In this paper we present an extractive summarization method for the Kazakh language based on fuzzy logic. We aimed to extract and concatenate important sentences from the primary text to obtain its shorter form. With the rapid growth of information on the Internet there is a demand on its efficient and cost-effective summarization.

**3240 Mohamed Ali Batita, Rami Ayadi, Mounir Zrigui** from Tunisia their paper “Reasoning over  
*Computación y Sistemas*, Vol. 23, No. 3, 2019, pp. 625–631  
doi: 10.13053/CyS-23-3-3208

Arabic WordNet Relations with Neural Tensor Network” ABSTRACT: Arabic WordNet is an important resource for many tasks of natural language processing. However, it suffers from many problems. In this paper, we address the problem of the unseen relationships between words in Arabic WordNet. More precisely, we focus on the ability for new relationships to be learned ‘automatically’ in Arabic WordNet from existing relationships.

**3241 Sanjay Kamath, Brigitte Grau, Yue Ma** from France in their paper “Predicting and Integrating Expected Answer Types into a Simple Recurrent Neural Network Model for Answer Sentence Selection” ABSTRACT: Since end-to-end deep learning models have started to replace traditional pipeline architectures of question answering systems, features such as expected answer types which are based on the question semantics are seldom used explicitly in the models.

**3242 Marek Menšík, Marie Duží, Adam Albert, Vojtěch Patschka, Miroslav Pajr** from Czech Republic in their paper “Refining Concepts by Machine Learning” ABSTRACT: In this paper we deal with machine learning methods and algorithms applied in learning simple concepts by their refining or explication. The method of refining a simple concept of an object *O* consists in discovering a molecular concept that defines the same or a very similar object to the object *O*. Typically, such a molecular concept is a professional definition of the object, for instance a biological definition according to taxonomy, or legal definition of roles, acts, etc.

**3243 Yingju Xia, Zhongguang Zheng, Yao Meng, Jun Sun** from China in their paper “Semi-automatic Knowledge Graph Construction by Relation Pattern Extraction” ABSTRACT: Knowledge graphs represent information in the form of entities and relationships between them. A knowledge graph consists of multi-relational data, having entities as nodes and relations as edges.

**3246 Amarnath Pathak, Ranjita Das, Partha Pakray, Alexander Gelbukh** from India in their paper “Extracting Context of Math Formulae contained inside scientific documents” ABSTRACT: A math formula present inside a scientific document is often preceded by its textual description, which is commonly referred to as the context of formula. Annotating context to the formula enriches its semantics, and consequently impacts the retrieval of mathematical contents from scientific documents.

**3247 Arda Akdemir, Tunga Gungor** from Japan and Turkey in her paper “Joint Learning of Named Entity Recognition and Dependency Parsing using Separate Datasets” ABSTRACT Joint learning of different NLP-related tasks is an emerging research field in Machine Learning. Yet, most of the recent models proposed on joint learning require a dataset that is annotated jointly for all the tasks involved.

**3248 Sunita Warjri, Partha Pakray, Saralin Lyngdoh, Arnab Kumar Maji** from India in their paper “Identification of POS Tag for Khasi Language based on Hidden Markov Model POS Tagger” ABSTRACT: Computational Linguistic (CL) becomes an essential and important amenity in the present scenarios, as many different technologies are involved in making machines to understand human languages.

**3249 Christopher G. Harris, Padmini Srinivasan** from United States in their paper “My Word! Machine versus Human Computation Methods for Identifying and Resolving Acronyms” ABSTRACT Acronyms are commonly used in human language as alternative forms of concepts to increase recognition, to reduce duplicate references to the same concept, and to stress important concepts.

**3250 Alejandra Segura Navarrete, Christian Vidal Castro, Clemente Rubio Manzano, Claudia Martínez Araneda** from Chile in his paper “The role of WordNet similarity in the affective analysis pipeline” ABSTRACT: Sentiment Analysis (SA) is a useful and important

discipline in Computer Science, as it allows having a knowledge base about the opinions of people regarding a topic. This knowledge is used to improve decision-making processes. One approach to achieve this is based on the use of lexical knowledge structures.

**3251 Rim Koulali, Abdelouafi Meziane** from Morocco in their paper “A Comparative Study on Text Representation Models For Arabic Topic Detection” ABSTRACT: Topic Detection (TD) plays a major role in Natural Language Processing (NLP). Its applications range from Question Answering to Speech Recognition. In order to correctly detect document’s topic, we shall first proceed with a text representation phase to transform the electronic documents contents into an efficiently software handled form.

**3252 Samarth Navali, Jyothirmayi Kolachalam, Vanraj Vala** from India in their paper “Sentence Generation Using Selective Text Prediction” ABSTRACT: Text generation based on comprehensive datasets has been a well-known problem from several years. The biggest challenge is in creating a readable and coherent personalized text for specific user. Deep learning models have had huge success in the different text generation tasks such as script creation, translation, caption generation etc.

**3253 K. Chakma, Amitava Das, Swapan Debbarma** from India in their paper “Deep Semantic Role Labeling for Tweets using 5W1H: Who, What, When, Where, Why and How” ABSTRACT . In natural language understanding, Semantic Role Labeling (SRL) is considered as one of the important tasks and widely studied by the research community. State-of-the-art lexical resources have been in existence for defining the semantic role arguments with respect to the predicates.

**3254 Rejwanul Haque, Mohammed Hasanuzzaman, Andy Way** from Ireland in their paper “Mining Purchase Intent in Twitter” ABSTRACT Most social media platforms allow

users to freely express their beliefs, opinions, thoughts, and intents. Twitter is one of the most popular social media platforms where users’ post their intent to purchase. A purchase intent can be defined as measurement of the probability that a consumer will purchase a product or service in future. Identification of purchase intent in Twitter sphere is of utmost interest as it is one of the most long-standing and widely used measures in marketing research.

**3255 Masaki Murata, Natsumi Morimoto** from Japan in their paper “Relation between Titles and Keywords in Japanese Academic Papers using Quantitative Analysis and Machine Learning” ABSTRACT In this study, we analyzed keywords from different academic papers using data from more than 300 papers. Using the concept of quantitative surveys and machine learning, we conducted various analyses on the keywords in different papers.

**3256 Sandra J. Gutiérrez Hinojosa, Hiram Calvo, Marco A. Moreno Armendáriz** from Mexico in their paper “Central Embeddings for Extractive Summarization Based on Similarity” ABSTRACT In this work we propose using word embeddings combined with unsupervised methods such as clustering for the multi-document summarization task of DUC (Document Understanding Conference) 2002.

**3257 Bharat Gaing, Nitish Varshney, Shubham Goel, Akash Mondal** from India in their paper “Identifying Short-Term Interests from Mobile App Adoption Pattern” ABSTRACT With the increase in an average user’s dependence on their mobile devices, the reliance on collecting user’s browsing history from mobile browsers has also increased. This browsing history is highly utilized in the advertising industry for providing targeted ads in the purview of inferring user’s short-term interests and pushing relevant ads.

**3258 Shubham Krishna, Ahsaas Bajaj, Mukund Rungta, Hemant Tiwari, Vanraj Vala** from India in their paper “RelEmb: A Relevance-based Application Embedding for Mobile App

Retrieval and Categorization” ABSTRACT Information Retrieval Systems have revolutionized the organization and extraction of Information. In recent years, mobile applications (apps) have become primary tools of collecting and disseminating information.

**3259 Jakub Sido, Miloslav Konopík, Ondřej Pražák** from Czech Republic in their paper “English Dataset for Automatic Forum Extraction” ABSTRACT This paper describes the process of collecting, maintaining and exploiting an English dataset of web discussions. The dataset consists of many web discussions with hand-annotated posts in the context of a tree structure of a web page.

**3260 Imran Sheikh, Balamallikarjuna Garlapati, Srinivas Chalamala, Sunil Kumar Kopparapu** from India in their paper “A Fuzzy Approach to Mute Sensitive Information in Noisy Audio Conversations” ABSTRACT Audio conversation between a service seeking customer and an agent are common in a voice based call center (VbCC) and are often recorded either for audit purposes or to enable the VbCC to improve their efficiency. These audio recordings invariably contain personal information of the customer, often spoken by the customer to confirm their identity to get personalized services from the agent.

**3263 Girish K. Palshikar, Sachin Pawar, Rajiv Srivastava, Mahek Shah** from India in their paper “Identifying Repeated Sections within Documents” ABSTRACT Identifying sections containing a logically coherent text about a particular aspect is important for fine-grained IR, question-answering and information extraction. We propose a novel problem of identifying repeated sections, such as project details in resumes and different sports events in the transcript of a news broadcast. We focus on resumes and present four techniques (2 unsupervised, 2 supervised) for automatically identifying repeated project sections.

**3264 Aлымzhan Toleu, Gulmira Tolegen, Rustam Mussabayev** from Kazakhstan in their paper “KeyVector: Unsupervised Keyphrase Extraction Using Weighted Topic via Semantic Relatedness” ABSTRACT . Keyphrase extraction is a task of automatically selecting topical phrases from a document. We present KeyVector, an unsupervised approach with weighted topics via semantic relatedness for keyphrase extraction.

**3265 Thuong-Hai Pham, Dominik Macháček, Ondřej Bojar** from Czech Republic in their paper “Promoting the Knowledge of Source Syntax in Transformer NMT” ABSTRACT The utility of linguistic annotation in neural machine translation has been already established. The experiments were however limited to recurrent sequence-to-sequence architectures and relatively small data settings.

**3266 Sapan Shah, Sarath S., Sreedhar Reddy** from India in their paper “Similarity Driven Unsupervised Learning for Materials Science Terminology Extraction” ABSTRACT Knowledge of material properties, microstructure, underlying material composition and manufacturing process parameters that the material has undergone is of significant interest to materials scientists and engineers.

**3267 Salima Harrat, Karima Meftouh, Kamel Smaili** from Algeria and France in their paper “Script Independent Morphological Segmentation for Arabic Maghrebi Dialects: An Application to Machine Translation” ABSTRACT This research deals with resources creation for under-resourced languages. We try to adapt existing resources for other resourced-languages to process less-resourced ones.

**3268 Lukas Svoboda, Tomas Brychcín** from Czech Republic in their paper “Enriching Word Embeddings with Global Information and Testing on Highly Inflected Language” ABSTRACT In this paper we evaluate our new approach based on the Continuous Bag-of-Words and Skip-gram models enriched with global context information on highly inflected Czech language and compare

it with English results. As a source of information we use Wikipedia, where articles are organized in a hierarchy of categories.

**3269 Rijul Dhir, Santosh Kumar Mishra, Sriparna Saha, Pushpak Bhattacharyya** from India in their paper “A Deep Attention based Framework for Image Caption Generation in Hindi Language” ABSTRACT Image captioning refers to the process of generating a textual description for an image which defines the object and activity within the image. It is an intersection of computer vision and natural language processing where computer vision is used to understand the content of an image and language modelling from natural language processing is used to convert an image into words in the right order.

**3270 Murillo Lagranha Flores, Elder Rizzon Santos, Ricardo Azambuja Silveira** from Brazil in their paper “Ontology-based Extractive Text Summarization: The Contribution of Instances” ABSTRACT In this paper, we present a text summarization approach focusing on multi-document, extractive and query-focused summarization that relies on an ontology-based semantic similarity measure, that specifically explores ontology instances.

**3271 Masahiro Kaneko, Mamoru Komachi** from Japan in their paper “Multi-Head Multi-Layer Attention to Deep Language Representations for Grammatical Error Detection” ABSTRACT It is known that a deep neural network model pre-trained with large-scale data greatly improves the accuracy of various tasks, especially when there are resource constraints. However, the information needed to solve a given task can vary, and simply using the output of the final layer is not necessarily sufficient.

**3272 Basma El Amel Boussaha, Nicolas Hernandez, Christine Jacquin, Emmanuel Morin** from France in their paper “Towards Simple but Efficient Next Utterance Ranking” ABSTRACT Retrieval-based dialogue systems converse with humans by ranking candidate responses according to their relevance to the history of the

conversation (context). Recent studies either match the context with the response on only sequence level or use complex architectures to match them on the word and sequence levels. We show that both information levels are important and that a simple architecture can capture them effectively.

**3273 Shashavali D., Vishwjeet, Rahul Kumar, Gaurav Mathur, Nikhil Nihal, Siddhartha Mukherjee, Suresh Venkanagouda Patil** from India in their paper “Sentence Similarity Techniques for short vs Variable Length Text using Word Embeddings” ABSTRACT In goal-oriented conversational agents like Chatbots, finding the similarity between user input and representative text result is a big challenge. Generally, the conversational agent developers tend to provide a minimal number of utterances per intent, which makes the classification task difficult.

**3274 Lucie Poláková, Jiří Mírovský** from Czech Republic in their paper “Anaphoric Connectives and Long-Distance Discourse Relations in Czech” ABSTRACT This paper is a linguistic as well as technical survey for the development of a shallow discourse parser for Czech. It focuses on long-distance discourse relations signalled by (mostly) anaphoric discourse connectives.

**3275 Dwijen Rudrapal, Amitava Das, Baby Bhattacharya** in their paper “A New Approach for Twitter Event Summarization Based on Sentence Identification and Partial Textual Entailment” ABSTRACT Recent trend of information propagation on any real-time event in Twitter makes this platform more and more popular than any other online communication media. This trend creates a necessity to understand real-time events quickly and precisely by summarizing all the relevant tweets. In this paper, we propose a two-phase summarization approach to produce abstract summary of any Twitter event.

**3276 Kushal Singla, Joy Bose, Nitish Varshney** from India in their paper “Word Embeddings for IoT Based on Device Activity

Footprints” ABSTRACT With the expansion of IoT ecosystem, there is an explosion of the number of devices and sensors and the data generated by these devices. However, the tools available to analyze such data are limited. Word embeddings, widely used in the natural language processing (NLP) domain, provides a way to get similar words to the current word.

**3277 Nadeesha Pathirana, Sandaru Seneviratne, Rangika Samarawickrama, Shane Wolff, Charith Chitraranjan, Uthayasanker Thayasivam, Tharindu Ranasinghe** from Sri Lanka in their paper “Concept Discovery through Information Extraction in Restaurant Domain” ABSTRACT Concept identification is a crucial step in understanding and building a knowledge base for any particular domain.

**3278 Ajay Nagar, Anmol Bhasin, Gaurav Mathur** from India in their paper “Text Classification using gated fusion of n-gram features and semantic features” ABSTRACT We introduce a novel method for text classification based on gated fusion of n-gram features and semantic features of the text. The parallel CNN network captures the n-gram relation between the words based on the filter size, pri-marily short distance multi-word relations.

**3279 Rodrigo López, Daniel Peñaloza, Francisco Beingolea, Juanjose Tenorio, Marco Sobrevilla Cabezudo** from Brazil in their paper “An Exploratory Study of the Use of Senses, Syntax and Cross-Linguistic Information for Subjectivity Detection in Spanish” ABSTRACT This work presents an exploratory study of Subjectivity Detection for Spanish This study aims to evaluate the use of dependency relations, word senses and cross-linguistic information in Subjectivity Detection task.

**3280 Lorenzo Porcaro, Horacio Saggion** from Spain in their paper “Recognizing Musical Entities in User-generated Content” ABSTRACT Recognizing Musical Entities is important for Music Information Retrieval (MIR) since it can improve the performance of several tasks such as

music recommendation, genre classification or artist similarity.

**3281 Maria Janicka, Maria Pszona, Aleksander Wawer** from Poland in their paper “Cross-Domain Failures of Fake News Detection” ABSTRACT Fake news recognition has become a prominent research topic in natural language processing. Researchers reported significant successes when applying methods based on various stylometric and lexical features and machine learning, with accuracy reaching 90%. This article is focused on answering the question: are the fake news detection models universally applicable or limited to the domain they have been trained on? We used four different, freely available English language Fake News corpora and trained models in both in-domain and cross-domain setting. We also explored and compared features important in each domain. We found that the performance in cross-domain setting degrades by 20% and sets of features important to detect fake texts differ between domains. Our conclusions support the hypothesis that high accuracy of machine learning models applied to fake news detection may be related to over-fitting, and models need to be trained and evaluated on mixed types of texts.

**3282 Mohammed Alliheedi, Robert E. Mercer, Sandor Haas Neil** from Canada in their paper “Ontological Knowledge for Rhetorical Move Analysis” ABSTRACT Scholarly writing in the experimental biomedical sciences follows the IMRaD (Introduction, Methods, Results, and Discussion) structure. Many Biomedical Natural Language Processing tasks take advantage of this structure. Recently, a new challenging information extraction task has been introduced as a means of obtaining these types of detailed information: identifying the argumentation structure in biomedical articles. Argumentation mining can be used to validate scientific claims and experimental methodology, and to plot deeper chains of scientific reasoning. One subtask in identifying the argumentation structure is the identification of rhetorical moves, text segments that are rhetorical and perform specific



communicative goals, in the Methods section. Based on a descriptive taxonomy of rhetorical moves structured around IMRaD, the foundational linguistic knowledge needed for a computationally feasible model of the rhetorical moves is described: semantic roles. One goal is to provide FrameNet and VerbNet-like ontologies for the specialized domain of biochemistry. Using the observation that the structure of scholarly writing in the laboratory-based experimental sciences closely follows the laboratory procedures, we focus on the procedural verbs in the Methods section. Occasionally, the text does not contain fillers for all of the semantic role slots

that are needed to perform an adequate analysis of a verb. To overcome this problem, an ontology of experimental procedures can be interrogated to provide a most likely candidate for the missing semantic role(s).

Alexander Gelbukh  
Guest Editor

Instituto Politécnico Nacional,  
Centro de Investigación en Computación,  
Mexico City, Mexico



# Ontological Knowledge for Rhetorical Move Analysis

Mohammed Alliheedi<sup>1,3</sup>, Robert E. Mercer<sup>2,3</sup>, Sandor Haas-Neill<sup>4</sup>

<sup>1</sup> Al Baha University, Faculty of Computer Science and Information Technology, Al Baha, Kingdom of Saudi Arabia

<sup>2</sup> The University of Western Ontario, Dept. of Computer Science, London, Canada

<sup>3</sup> University of Waterloo, David R. Cheriton School of Computer Science, Waterloo, ON, Canada

<sup>4</sup> McMaster University, Dept. of Medical Science, Hamilton, ON, Canada

malihee@uwaterloo.ca, mercer@csd.uwo.ca, haasneis@mcmaster.ca

**Abstract.** Scholarly writing in the experimental biomedical sciences follows the IMRaD (Introduction, Methods, Results, and Discussion) structure. Many Biomedical Natural Language Processing tasks take advantage of this structure. Recently, a new challenging information extraction task has been introduced as a means of obtaining these types of detailed information: identifying the argumentation structure in biomedical articles. *Argumentation mining* can be used to validate scientific claims and experimental methodology, and to plot deeper chains of scientific reasoning. One subtask in identifying the argumentation structure is the identification of *rhetorical moves*, text segments that are rhetorical and perform specific communicative goals, in the Methods section. Based on a descriptive taxonomy of rhetorical moves structured around IMRaD, the foundational linguistic knowledge needed for a computationally feasible model of the rhetorical moves is described: *semantic roles*. One goal is to provide FrameNet and VerbNet-like ontologies for the specialized domain of biochemistry. Using the observation that the structure of scholarly writing in the laboratory-based experimental sciences closely follows the laboratory procedures, we focus on the procedural verbs in the Methods section. Occasionally, the text does not contain fillers for all of the semantic role slots that are needed to perform an adequate analysis of a verb. To overcome this problem, an ontology of experimental procedures can be interrogated to provide a most likely candidate for the missing semantic role(s).

**Keywords.** Ontological knowledge, rhetorical move.

## 1 Introduction

Scientists must routinely review the scholarly literature in their fields to keep abreast of current advances and to retrieve information relevant to their research. However, the volume of online scientific literature is immense, and rapidly increasing. In the biomedical field, the National Centre for Biotechnology Information (NCBI) developed a literature search engine, PubMed<sup>1</sup>, to access various databases such as MEDLINE (journal citations and abstracts for biomedical literature), full-text life science e-journals, and online books.

In 2010 PubMed repositories consisted of more than 20 million citations for biomedical literature [23]. By 2019 the number of citations had increased to more than 30 million<sup>2</sup>. As a consequence, it has become extremely challenging for biomedical scientists to keep current with information in their fields. This challenge has attracted Natural Language Processing (NLP) researchers to develop resources and automated tools for performing various tasks in Information Extraction (IE) and Text Mining (TM) using online corpora of biomedical articles, and thus enable

<sup>1</sup><http://www.ncbi.nlm.nih.gov/pubmed>

<sup>2</sup><http://www.ncbi.nlm.nih.gov/books/NBK3827>

biomedical researchers to better manage and exploit this volume of data [18].

These research activities have led to the development of a new field, Biomedical Natural Language Processing (BioNLP), a collaboration between the biomedical and computational linguistics/artificial intelligence communities [17]. The types of tasks currently handled by BioNLP systems have generally been aimed at extracting very specific and limited information, for example, protein and gene names and relations [11], and so have been able to rely on relatively simple forms of information extraction. BioNLP has adapted various standard information extraction techniques, including both rule-based (e.g., shallow parsing, syntactic pattern-matching) and Machine Learning (e.g., Support Vector Machines, k-nearest neighbour classification method), to address several text-mining tasks, including extracting: protein-protein interactions (PPI) [21], drug-drug interactions (DDI) [28], gene relationships [19], and protein-residue associations [25].

Although these approaches fulfil some information needs, information extraction systems based on these can only recognize and extract minimal and specific information from biomedical texts. But other, more in-depth and comprehensive, information contained in biomedical texts would be highly valuable to scientists because this type of information can enable validating scientific claims, tracing current research directions in their field, reproducing scientific procedures and so forth. Recently, a new and more challenging information extraction task has been introduced as a means of obtaining these types of detailed information: identifying the argumentation structure in biomedical articles (e.g., [15] and [16]). *Argumentation mining* can be used to validate scientific claims and experimental methodology, and to plot deeper chains of scientific reasoning. Unlike earlier simpler forms of information extraction, here the goal is to identify the structure of argumentative components within an entire text—for example, premises, evidence, conclusions—as well as the relationships between components.

To achieve this goal the text needs to be analyzed. Our approach to this analysis is based on a working hypothesis:

We hypothesize that recognizing and detecting rhetorical moves would provide important information to our argumentation analysis framework, and that the Method sections in biochemistry articles contain moves which can be correlated with the author's experimental procedures. These moves can be used to determine salient information about the elements of the article's argumentative structure (e.g., premises) and can contribute to the overall understanding of the author's scientific claims.

A key aspect of our hypothesis is that development of a frame-based knowledge representation can be based on the semantics of the verbs associated with these procedures. This representation can provide detailed knowledge for understanding these rhetorical moves, which will in turn facilitate analysis of argumentation structure. In other words, we propose that a *procedurally rhetorical verb-centric frame semantics* can be used to obtain a sufficiently deep analysis of sentence meaning.

While this approach seems straightforward enough, the writing style of biochemistry articles requires the reader to have knowledge about biochemistry and biochemistry laboratory techniques and practices. This paper first gives the semantic roles that can be used in the semantics of each verb. Then an example of how an ontology containing knowledge about biochemistry laboratory techniques and practices can be used to fill the semantic roles of verbs which cannot be filled by information in the text.

## 2 Related Work

Swales [29] proposed the Create-A-Research-Space (CARS) model that uses intuition about the argumentative structure of scientific research articles. Swales defined rhetorical moves as text segments that convey communicative goals. He reviewed the Introduction section in 48 articles from social and natural science and found common rhetorical structures among most of these articles. Swales identified three moves in these articles: establishing a research territory, establishing a niche, and occupying the niche.

However, despite the widespread influence of the CARS model, some researchers observed two problems: (i) the inconsistent assignment of rhetorical moves to text segments because the identification of the rhetorical moves relies on overall text comprehension, and (ii) a lack of empirical validation of moves in linguistic terms [20].

To overcome these problems, Kanoksilapatham [20] advanced Swales' approach to move analysis by developing a framework that combines his original CARS model with the use of Biber's multidimensional analysis [6] to enrich the model with additional information about linguistic characteristics. Biber's multidimensional analysis [6] is concerned with variation in the speaking and writing of English. Multidimensional analysis can be used to identify differences in linguistic characteristics between various text types at different levels of document structure (e.g., genre, internal section level). Although Kanoksilapatham provides an extension to the Swales's move analysis study, and attempted validation of these moves in biochemistry articles, she only provides a descriptive analysis about rhetorical moves without defining an explicit method for analyzing and recognizing these moves in texts.

Liakata et al. [22] developed an annotation scheme called Core Scientific Concepts (CoreSC) to classify sentences into scientific categories (e.g., related to author's other work). The CoreSC scheme consists of three layers: the first includes several categories to classify sentences; the second layer is concerned with properties of these categories; and the third layer creates a link to related instances of the same category. The authors use Machine Learning classifiers (i.e., Conditional Random Fields and Support Vector Machines) to automatically classify sentences into the CoreSC categories. The data set consisted of 265 biochemistry and chemistry articles. The authors were only able to achieve an accuracy around 50% in categorizing sentences in the appropriate CoreSC scientific categories which is inadequate for such a task.

Green [15] proposed a plan for creating an annotated corpus of biomedical genetics research articles. Green emphasized that this corpus

would be beneficial to the argumentation mining community since it would provide a fine-grained annotation of argumentative components. Also since there are as yet few annotated corpora available, such a corpus would enrich research in the field of Computational Argumentation in general. The author stated that this corpus will be publicly available for further investigation by different research groups in various tasks of argumentation mining.

Green [16] specified a set of argumentation schemes for scientific claims in genetics research articles. The author used a corpus of unannotated genetics research articles, and identified the components (e.g., premises, conclusions) of an argument as well as its type of scheme. Based on the analyses of various genetics research articles, the author specified 10 argumentation schemes that are semantically different. These schemes were new and had not previously been proposed.

Furthermore, the specification of argumentation schemes was used to create annotation guidelines. Then, these guidelines were evaluated in a pilot study based on participants' ability to recognize these schemes by reading the guidelines. Overall, the author's ultimate goal for this initial study was to develop annotation guidelines for creating corpora for argumentation mining research. However, based on the pilot study, the results showed a variation in performance since there were two groups of participants (i.e., undergraduate students and researchers). The students performed poorly in recognizing argumentation schemes while the researchers were able to identify these schemes correctly in most cases.

### 3 Our Proposed Approach: Rhetorical Moves Mirror Scientific Experimental Procedures

Our intention is to develop a formal knowledge representation based on procedural verbs as a method for argumentation analysis. We introduced the notion of Swale's CARS model [29] in Section 2. We hypothesize that recognizing and detecting rhetorical moves would provide additional information to our framework of argumentation

analysis. We also hypothesize that the Method sections in biochemistry articles contain moves which can be correlated with the author's experimental procedures. These moves can be used to determine salient information about the elements of the article's argumentative structure (e.g., premises) and can contribute to the overall understanding of the author's scientific claims. A key aspect of our hypothesis is that development of a frame-based knowledge representation can be based on the semantics of the verbs associated with these procedures. This representation can provide detailed knowledge for understanding these rhetorical moves, which will in turn facilitate analysis of argumentation structure. In other words, we propose that *procedurally rhetorical verb-centric frame semantics* can be used to obtain a deeper analysis of sentence meaning than is currently the case with simple methods of Information Extraction (e.g., shallow syntactic pattern) and in a computationally feasible manner.

Scientific argument<sup>3</sup> is defined as a process that scientists follow by using certain procedures to obtain empirical data which will either support or defeat their claims, hence leading to the intended conclusion. The strength of a scientific argument depends on its reproducibility and consistency. For a scientific argument to be strong, a scientist should identify and explain all the procedures in their experiment, i.e., reproducibility, so that another researcher who follows the same procedures will reach the same conclusion, i.e., consistency. Thus, for a well-constructed scientific article, a scientist should expect the same conclusion if she follows the same procedures in the same sequence as described in the Method section.

Scientific writing in the biochemistry domain has certain characteristics that made it ideal for our purposes. In this domain, experimental procedures describe the sequence of actions the biochemist performs to carry out an experiment to derive verifiable scientific conclusions. The experimental procedures themselves can be verified because

they are standard procedures described in detail in experimental manuals (e.g., Boyer [7] and Sambrook and Russell [26]). Verbs play an essential role as indicators of these experimental procedures.

These procedures can be viewed as corresponding to the elements of the scientific argumentation structure. For example, when examining a biological substance (e.g., a certain type of bacteria) in order to prove a hypothesis (e.g., this bacteria is correlated with a certain disease) the biochemist would perform a sequence of certain procedures to arrive at a conclusion. Essentially, biochemists create an argumentation framework through the scientific methodology they follow—how they perform their experiments is how they argue. We can observe that this genre—biochemistry articles—is procedure-oriented since the scientific procedures that are described are parallel to the scientific argumentation in the text. For example:

**Example 1** *"Beads with bound proteins were washed six times (for 10 min under rotation at 4 C) with pulldown buffer and proteins harvested in SDS-sample buffer, separated by SDS-PAGE, and analyzed by autoradiography."* [12].

In this example, the verbs "washed", "harvested", "separated", and "analyzed" are used to illustrate the procedure steps in sequential order. Such an experiment can be reproduced if one follows these steps.

Fillmore [13] introduced the notion of frame semantics as a theory of meaning. A *semantic frame* is defined as "any coherent individuable perception, memory, experience, action or object" by Fillmore [14]. In other words, coherently structured concepts that are related to each other represent a complete knowledge of world events or experiences. For example, to understand the word "buy", one would access the knowledge contained in the commercial transaction frame which includes words such as the person who buys the goods (buyer), the goods that are being sold (goods), the person who sells the goods (seller), and the currency that the buyer and seller agree on (money).

<sup>3</sup><http://www.ces.fau.edu/nasa/introduction/scientific-inquiry/why-do-scientists-argue-and-challenge-each-others-results.php>

Following Fillmore's theory of frame semantics, FrameNet [5] was developed to create an online lexical resource for English. This framework includes more than 170,000 manually annotated sentences and 10,000 words. The computational linguistic community has been attracted to the concept of the frame semantics and developed computational resources using this concept, such as VerbNet [27], an on-line verb lexicon for English and PropBank [24], an annotated corpus with basic semantic propositions.

Following the notion of frame semantics, we propose to build a knowledge representation framework to analyze verbs in a procedure-oriented genre. Our concept of procedurally rhetorical verb-centric frame semantics is intended to address this gap by developing a computationally feasible knowledge representation that will enable argumentation analysis.

The knowledge contained in the frame semantics will facilitate the extraction of elements of arguments, i.e., argumentation mining. To reiterate, our hypothesis is that procedurally rhetorical verb-centric frame semantics can provide a knowledge representation framework for analyzing and representing the meanings of the verbs used in biochemistry articles. In turn, these frames will facilitate the identification of argumentation structure in the discourse describing experimental procedures.

## 4 Ontological Knowledge Sources

To provide the knowledge required to achieve the rhetorical move analysis discussed in the previous section, we propose two sources organized as ontologies. An ontology, as used here, is composed of the concepts and the relations between them. We discuss two ontologies below. The first, semantic roles, represents the knowledge about verbs that we argue is needed to analyze rhetorical moves. This information is organized in VerbNet-like [27] verb frames. The second knowledge source is composed of information about experimental procedures in the biochemistry domain. This information is organized in the familiar graph-based web of objects, classes of objects, and relations among these.

### 4.1 Semantic Roles

As described earlier our experimental event scheme was inspired by the annotation scheme for bio-events [30]. We based our experimental event scheme for verb arguments on the inventory of semantic roles in VerbNet [27] and modified and added new semantic roles to define our scheme. Our experimental event scheme includes: *Theme*, *Patient*, *Predicate*, *Agent*, *Location*, *Goal*, etc. The complete set of semantic roles and their definitions in our experimental event scheme is presented in Table 2.

We have extended the VerbNet definition of the semantic role *Instrument* from simply describing "an object or force that comes in contact with an object and causes some change in them" [27] to include a variety of subcategories that correspond to various types of biological and man-made instruments that are used in a biochemistry laboratory. The new semantic roles (with example text in boldface) are:

1. Instruments used to *change* the state of an object. For example:

**Example 2** "Beads with bound proteins were washed six times (for 10 min under rotation at 4 C) with **pulldown buffer** ..." [12].

In this example, the pulldown buffer was used to wash (change the state of) the Beads with bound proteins. In this instance, the phrase "pulldown buffer" should be labeled as **instrument (change)**.

2. Instruments used to *maintain* the state of an object. For example:

**Example 3** "Once the samples were in EPR tubes, they were immediately frozen in liquid nitrogen, and stored in **liquid nitrogen** before using." [10].

In this example, the liquid nitrogen was used to store (maintain the condition of) the samples which were in the EPR tubes. In this case, the phrase "liquid nitrogen" should be labeled as **instrument (maintain)**.

**Table 1.** Rhetorical Moves in the Method Section of Biochemistry Articles (from [20])

Move type	Definition
Description-of-method	Concerned with sentences that describe experimental events.
Appeal-to-authority	Concerned with sentences that discuss the use of well-established methods.
Background information	Concerned with all background information for the experimental events such as “method justification, comment, or observation, exclusion of data, approval of use of human tissue” as defined by Kanoksilapatham (2003).
Source-of-materials	Concerned with the use of certain biological materials in the experimental events.

3. Instruments used to *observe* an object. For example:

**Example 4** *The mitochondria was observed by spinning disk confocal microscopy.*

The spinning disk confocal microscopy is used to observe the mitochondria. We should label the phrase “spinning disk confocal microscopy” as **instrument (observe)**.

4. Instruments used as a *catalyst* in experimental processes to occur. For example:

**Example 5** *“The ca. 900 bp PCR products were digested with **NdeI and HindIII** and ligated into pUC19.” [9].*

In this example, the *NdeI* and *HindIII* are enzymes used to facilitate the digestion (cutting) of the ca.(approximately) 900 bp PCR products. In this instance, the phrase “*NdeI* and *HindIII*” should be labeled as **instrument (catalyst)**.

5. Instrument used to *measure* an object. For example:

**Example 6** *“Beads with bound proteins were washed six times (for 10 min under rotation at 4 C) with pulldown buffer and proteins harvested in SDS-sample buffer, separated by SDS-PAGE, and analyzed by **autoradiography**.” [12].*

In this example, the autoradiography was used to analyze (measure) the proteins. In this example, the word “autoradiography” should be labeled as **instrument (measure)**.

6. It could be used to describe a *mathematical or computational instrument* (e.g., simulation, algorithm, equation, and the use of software). For example:

**Example 7** *“Simulations of these EPR spectra were accomplished with **the computer program QPOWA** [30,31].” [10].*

The computer program QPOWA was used here as computational instrument to perform simulations of the mentioned above EPR spectra. So, the phrase “the computer program QPOWA [30,31]” should be labeled as **instrument (computational instrument)**.

7. Finally it could be used as a *reference* for method or protocol that being used. For example:

**Example 8** *“The preparation of authentic vaccinia H5R protein and recombinant B1R protein kinase were **as previously described** [11].” [8]*

The phrase “as previously described [11]” is to indicate that the authors referring to other method that they used in their current experimental process. We should label the phrase “as previously described [11]” as **instrument (reference)**.

These sub-categories of the semantic role (instrument) are not necessarily exclusive to the mentioned types above. However, based on our full-text analysis, these instrument types are as comprehensive as we have achieved to date. We



**Table 2.** Semantic Roles in the Annotation Scheme of our Experimental Event

Semantic role	Definition
Agent	Generally a human or an animate subject.
Patient	Participants that have undergone a process.
Theme	Participants in a location or undergoing a change of location.
Goal:	
Physical	Identifies a thing toward which an action is directed or a place to which something moves.
Purpose	Identifies the stated purpose in a sentence for doing certain actions.
Factive	A referent that results from the action or state identified by a verb.
Location	The physical place where the experiments took place.
Protocol-Detail:	
Time	Identifies the time or a duration of an experimental process.
Temperature	Identifies the temperature of an experimental process.
Condition	Identifies the condition of how an experimental process is performed.
Repetition	Identifies the number of times an experimental process is repeated.
Buffer	Identifies the buffer that was used in an experimental process.
Cofactor	Identifies the cofactor that was used in an experimental process.
Instrument:	
Change	Describes objects (or forces) that come in contact with an object and cause some change.
Measure	Describes an object or protocol that can measure another object(s).
Observe	Describes an object which can be used to observe another object(s).
Maintain	Describes an object or protocol which can be used to maintain the state of object(s).
Catalyst	Describes an object that can be used as a catalytic “facilitator” for an experimental event to occur.
Reference	Refers to a method or protocol that is being used.
Mathematical	Describes a mathematical or computational instrument

will add or update these sub-categories if we encounter a new type (usage) of instrument.

We have also proposed a new semantic role *protocol detail* that identifies certain types of information about experimental processes. These new subcategories (with example text in boldface) are:

1. Time or the duration of a process [27]. For example:

**Example 9** “Beads with bound proteins were washed six times (**for 10 min** under rotation at 4 C) with pulldown buffer . . .” [12].

2. Temperature of an experimental process. For example:

**Example 10** “Beads with bound proteins were washed six times (for 10 min under rotation **at 4 C**) with pulldown buffer . . .” [12].

3. Condition or manner of which an experimental process was carried out. For example:

**Example 11** “Beads with bound proteins were washed six times (for 10 min **under rotation** at 4 C) with pulldown buffer . . .” [12].

4. Buffer which is “a solution containing either a weak acid and a conjugate base or a weak

FRAMES for digest-121
NP V PP
[NP Array-generated oligos ] [VP were digested ] [PP with ] [NP restriction enzymes ] ( [NP NotI ] and [NP EcoRI ] )
Example: “Array-generated oligos were digested with restriction enzymes (NotI and EcoRI)”
Syntax: PATIENT V INSTRUMENT
Semantics: MANNER (DURING(E), PATIENT)
NP V ADVP PP PP
[NP fractions ] [PP of ] [NP interest ] [VP containing ] [NP NS DNA ] [VP were digested ] [ADVP twice ] [PP with ] [NP λ-Exo ] [VP to eliminate ] [NP contaminating DNA ]
Example: “fractions of interest containing NS DNA were digested twice with λ-Exo to eliminate contaminating DNA.”
Syntax: PATIENT V REPETITION INSTRUMENT GOAL
Semantics: MANNER (DURING(E), PATIENT)
PP NP V PP PP
[ADVP Moreover ] , [PP as ] [NP a negative control ] [PP in ] [NP the above study ] , [NP a large amount ] [PP of ] [NP total fragmented DNA ] ( [NP 150 µg ] ) [VP was digested ] [PP with ] [NP λ-Exo ] [PP in ] [NP strong limiting conditions ] ( [NP 0.7 units ] [PP of ] [NP λ-exo /µg ] [PP of ] [NP DNA ] )
Example: “Moreover, as a negative control in the above study, a large amount of total fragmented DNA (150 µg) was digested with λ-Exo in strong limiting conditions (0.7 units of λ-exo /µg of DNA).”
Syntax: GOAL PATIENT V INSTRUMENT CONDITION
Semantics: MANNER (DURING(E), PATIENT)
NP V VP PP PP PP
[NP Forty micrograms ] [NP total RNA ] [VP was digested ] [VP using ] [NP 0.1 U nuclease P1 ] ( [NP Yamasa Corporation ] ) [PP in ] [NP 25 mM NH <sub>4</sub> OAc ] ( [NP pH 5.3 ] ) [PP at ] [NP 37 °C ] [PP for ] [NP 1 h ]
Example: “Forty micrograms total RNA was digested using 0.1 U nuclease P1 (Yamasa Corporation) in 25 mM NH <sub>4</sub> OAc (pH 5.3) at 37 °C for 1 h.”
Syntax: PATIENT V INSTRUMENT BUFFER TEMP TIME
Semantics: MANNER (DURING(E), PATIENT)
NP V PP VP PP
[NP DNA ] [VP was digested ] [PP with ] [NP HindIII restriction enzyme ] [VP leaving ] [NP an overhang ] [NP that ] [VP is filled ] [PRT in ] [PP by ] [NP biotinylated dCTP ]
Example: “DNA was digested with HindIII restriction enzyme leaving an overhang that is filled in by biotinylated dCTP.”
Syntax: PATIENT V INSTRUMENT FACTITIVE CONDITION
Semantics: MANNER (DURING(E), PATIENT)

Fig. 1. The verb frame for the verb *digest*

base and a conjugate acid, used to stabilize the pH of a liquid upon dilution.”<sup>4</sup> For example:

**Example 12** “For phosphorylation, three identical reactions contained H5R protein (70 pmol), B1R protein kinase (90 µl), **Tris-HCl, pH 7.4 (20 mM)**, magnesium chloride (5 mM), ATP (50 µM), [γ-32P] ATP (50 µCi) and dithiothreitol (2 mM) in a total volume of 500 µl” [8].

<sup>4</sup>Buffer - Biology-Online Dictionary. (n.d.). Retrieved September 23, 2017, from <http://www.biologyonline.org/dictionary/Buffer>

5. Cofactor is defined as “substances that are required for, or increase the rate of, catalysis.”<sup>5</sup> For example:

**Example 13** “For phosphorylation, three identical reactions contained H5R protein (70 pmol), B1R protein kinase (90 µl), **Tris-HCl, pH 7.4 (20 mM)**, **magnesium chloride (5 mM)**, **ATP (50 µM)**, [γ-32P] ATP (50 µCi) and

<sup>5</sup>coenzymes and cofactors. (n.d.). Retrieved September 23, 2017, from [http://academic.brooklyn.cuny.edu/biology/bio4fv/page/coenzy\\$\\\_\\$.htm](http://academic.brooklyn.cuny.edu/biology/bio4fv/page/coenzy$\_$.htm)

**Alkaline Agarose Gel Electrophoresis****1. Materials**

- 1.1. 10x Alkaline agarose gel electrophoresis buffer
- 1.2. 1x TAE electrophoresis buffer
- 1.3. 6x Alkaline gel-loading buffer
- 1.4. DNA samples (usually radiolabeled)
- 1.5. Agarose
- 1.6. DNA staining solution
- 1.7. Ethanol
- 1.8. Neutralizing solution for alkaline agarose gels
- 1.9. Sodium acetate (3 M, pH 5.2)

**2. Method****2.1. Prepare the agarose solution**

- 2.1.1. adding the appropriate amount of powdered agarose to a measured quantity of H<sub>2</sub>O in either:
  - an Erlenmeyer flask
    - Loosely plug the neck of the Erlenmeyer flask with Kimwipes
    - Container 1
  - or a glass bottle
    - make sure that the cap is loose
    - Container 1
- 2.1.2. Heat the slurry (Item 1) in (Container 1) for the minimum time required to allow all of the grains of agarose to dissolve using either:
  - a boiling-water bath
    - Check that the volume of the solution (Item 1) has not been decreased by evaporation during boiling in (Container 1);

1

- 2.3.1. Load the DNA samples dissolved in 6x alkaline gel-loading buffer into the wells of the gel (container 3)

- 2.3.2. Start the electrophoresis at <3.5 V/cm

- when the bromocresol green has migrated into the gel approx. 0.5-1 cm
  - turn off the power supply
  - and place a glass plate on top of the gel in (Container 3)
- Continue electrophoresis until:
  - the bromocresol green has migrated approximately two thirds of the length of the gel in (container 3).

**2.4. Finalize the experiment**

- 2.4.1. Process the gel according to one of the procedures either:

- Southern hybridization
  - Transfer the DNA either:
    - Directly (without soaking the gel) from the alkaline agarose gel to:
      - a charged nylon membrane
    - OR after soaking the gel in neutralizing solution for 45 minutes at room temperature to either:
      - an uncharged nitrocellulose.
      - or nylon membrane
    - As described in Southern Blotting: Capillary Transfer of DNA to Membranes
    - Please see Southern Blotting: Capillary Transfer of DNA to Membranes
  - Detect the target sequences in the immobilized DNA by hybridization to an appropriate labeled probe.
  - Please see Southern Hybridization of Radiolabeled Probes to Nucleic Acids Immobilized on Membranes

3

- if yes: replenish with H<sub>2</sub>O in (Container 1)
- If no: do not add H<sub>2</sub>O in (Container 1)

- or a microwave oven

- Check that the volume of the solution (Item 1) has not been decreased by evaporation during boiling in (Container 1);

- if yes: replenish with H<sub>2</sub>O in (Container 1)
- If no: do not add H<sub>2</sub>O in (Container 1)

- 2.1.3. Cool the clear solution (Item 1) to 55°C.

- Add 0.1 volume of 10x alkaline agarose gel electrophoresis buffer in (Container 1)

- and immediately pour the gel (Item 1) into mold (Container 2)

- 2.1.4. After the gel (Item 1) is completely set

- mount it (Item 1) in the electrophoresis tank (Container 3)

- add freshly made 1x alkaline electrophoresis buffer until the gel (Item 1) is just covered.

**2.2. Prepare DNA samples**

- 2.2.1. Collect the DNA samples (Item 2) by standard precipitation with ethanol

- 2.2.2. Dissolve the damp precipitates of DNA (Item 2) in 10-20 µl of 1x gel buffer. (Item 3)

- 2.2.3. Add 0.2 volume of 6x alkaline gel-loading buffer.

- 2.2.4. It is important to chelate all Mg<sup>2+</sup> with EDTA before adjusting the electrophoresis samples to alkaline conditions.

**2.3. Initiate the electrophoresis**

2

- or Staining

- Soak the gel in neutralizing solution for 45 minutes at room temperature.
- Stain the neutralized gel with 0.5 µg/ml ethidium bromide in 1x TAE or with SYBR Gold.
  - A band of interest can be sliced from the gel and subsequently eluted by one of the procedures described in the following protocol:
    - Recovery of DNA from Agarose Gels: Electrophoresis onto DEAE-cellulose Membranes or Recovery of DNA from Agarose and Polyacrylamide Gels: Electroelution into Dialysis Bags.

4

**Fig. 2.** Alkaline Agarose Gel Electrophoresis Ontology

**Table 3.** Some sentences from the article **Biochem-3--77373** [9]

No.	Sentence
1	The over-expression plasmid for L1, pUB5832, was digested with <i>NdeI</i> and <i>HindIII</i> , and the resulting ca. 900 bp piece was gel purified and ligated using T4 ligase into pUC19, which was also digested with <i>NdeI</i> and <i>HindIII</i> , to yield the cloning plasmid pL1PUC19.
2	Mutations were introduced into the L1 gene by using the overlap extension method of Ho et al. [60], as described previously [68].
3	The oligonucleotides used for the preparation of the mutants are shown in Table 1.1.

*dithiothreitol (2 mM) in a total volume of 500  $\mu$ l.*" [8].

6. Repetition of a step in experimental processes. For example:

**Example 14** "*Beads with bound proteins were washed **six times** (for 10 min under rotation at 4 C) with pulldown buffer . . .*" [12].

With these semantic roles we are able to provide the frames for procedural verbs. To illustrate, Fig. 1 contains the frame for the verb *digest*.

#### 4.2 An Ontology of Biochemical Techniques and Laboratory Practices

Knowledge about how experiments are carried out in a biochemistry laboratory is absolutely essential to the understanding of much of the text found in biochemistry articles. We needed assistance from a biochemist to understand many of the sentences that are present in our corpus. With this in mind we have developed an ontology prototype to assist with a computational approach to analyzing the sentences found in the Methods section of a biochemistry article. Details of this prototype ontology are described elsewhere [3].

The example of a procedure called Alkaline Agarose Gel Electrophoresis is given in text format in Fig. 2. This is a common procedure used to isolate the biological substance that is used in future procedures from the other substances found in the solution that results from the previous procedures. The knowledge about how this electrophoresis procedure is carried out has been

implemented in the prototype ontology. Why this knowledge is important is discussed in the following section.

#### 5 A Manual Annotation of a Portion of a Method Section

We have selected three articles from our corpus randomly to manually analyze and extract steps in experimental procedures (processes) from the method section. Table 3 shows some sentences from one of these articles [9]. The purpose of this analysis is to identify the semantic roles of experimental processes and the semantic frames of procedural verbs that occurred in these processes. Also, we want to demonstrate the usefulness of our approach by mapping the knowledge of frame semantics and the ontological knowledge to rhetorical moves.

The sentences in Table 3 are three contiguous sentences in a biochemistry article. They discuss the idea of cutting a DNA piece from a plasmid, which is "a small circular and double-stranded DNA molecule that is distinct from a cell's chromosomal DNA",<sup>6</sup> and ligate (attach) that piece to another plasmid to produce the desired protein. Table 4 shows five events from the sentences in Table 3. The events 1, 2, 3, and 4, which are demonstrated in Fig. 3, are extracted from Sentence No. 1, and Sentence No. 2 has only Event 5, while there is no actual experimental event in Sentence No. 3. It rather simply refers to a table in the article's

<sup>6</sup>plasmid / plasmids — Learn Science at Scitable. (n.d.). Retrieved December 22, 2017, from <https://www.nature.com/scitable/definition/plasmid-plasmids-28>

**Table 4.** Extracted events from two sentences in the article **Biochem-3-77373** [9]

Event 1	Event 2	Event 3
Sentence No. 1 — Patient: The over-expression plasmid for L1, pUB5832 — Predicate: digested — Instrument (catalyst): <i>NdeI</i> and <i>HindIII</i>	Sentence No. 1 — Patient: the resulting ca. 900 bp piece — Predicate: gel purified — Instrument (catalyst): Gel electrophoresis	Sentence No. 1 — Patient: pUC19 — Predicate: digested — Instrument (catalyst): <i>NdeI</i> and <i>HindIII</i>
Event 4	Event 5	
Sentence No. 1 — Patient: the resulting ca. 900 bp piece — Predicate: ligated — Instrument (catalyst): using T4 ligase — goal: into pUC19	Sentence No. 2 — Patient: the L1 gene — Predicate: introduced (mutated) — Instrument (reference type): using the overlap extension method of Ho et al.	Sentence No. 3 does not contain experimental events.

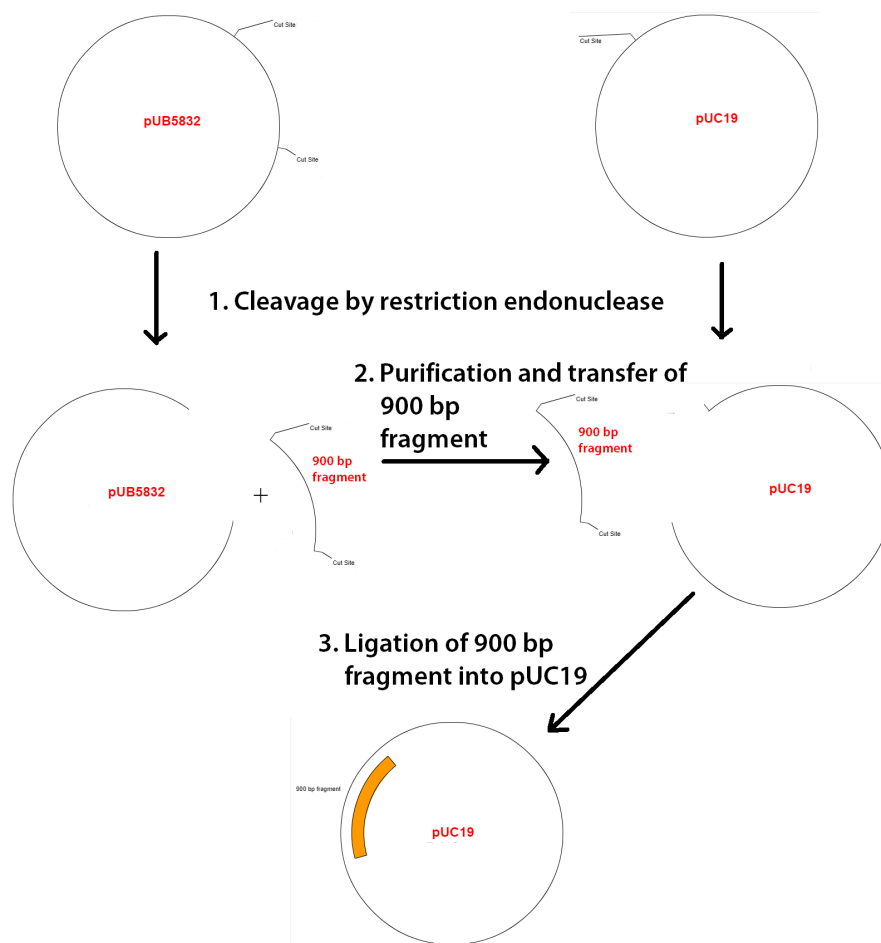
prior text. Each event in Table 4 represents one complete experimental procedure. Also the actual sequence of experimental events in the lab don't necessarily follow the sequence that these events appear in the text. Another important aspect to note is that not all the essential information about experimental processes is found in the text, some information can be implied. However, these implied pieces of information can be inferred from an ontology of standard biochemistry procedures, some of which we have developed. Taking a look at Events 1-4 in Table 4:

1. Digestion of pUB5832: a 900 bp piece was cut out using two restriction enzymes (*NdeI* and *HindIII*).
2. Then, the gel purification of the 900 bp piece: gel electrophoresis was used in this purification step. This is implied information derived from the ontology.
3. At any time before Event 4, the digestion of pUC19 happens, This could happen before, after, between, or during Events 1 and 2.

4. After Events 1, 2, and 3, ligation of the 900 bp into pUC19 occurs.

A lot of information can be derived from the text using knowledge about the verbs. This has been described earlier: the semantic roles of each verb together with syntactic information allows this information to be extracted from the text. Table 4 shows this extracted information. However, this is not enough to understand the information provided in the text.

A proper interpretation of the description of events in Sentence No. 1 cannot be completely derived from the text alone. An understanding of laboratory practice together with knowledge of what is involved in performing plasmid digestion, purification, and ligation is required. Some of the event sequencing can be derived from the text, for instance, the pragmatics of the conjunction “and” usually indicates that the second conjunct follows temporally after the first conjunct has completed. The phrase “the resulting” is also a key linguistic clue to determine this sequence. But, when the third event happens requires knowledge of



**Fig. 3.** A sequence of the events 1,2,3 and 4 from sentence No.1

biochemistry and laboratory practice as well as knowledge of the complete method. The linguistic information provided by the use of a relative clause does not enable a complete understanding of this event, so the ontology is required for the information required to do a proper interpretation. Another important aspect of the text is that all of the referents are described by singular nouns. However, knowing the biological processes that are carried out in the laboratory is important: solutions containing large numbers of the biological elements are used. Hence, one is not dealing with a single plasmid or a single piece from the plasmid,

and when the digestion occurs, all of the pieces from the plasmids are in the solution including ones that didn't get digested, thus the need for the gel purification step which separates the various biological elements.

An example of inferring implied information from the ontology can be given. Event 2 in Table 4 is gel purification. What is used to perform this task is not given in the text. The following SPARQL query extracts some domain knowledge about the experimental procedure of Alkaline Agarose Gel Electrophoresis from our framework providing the missing instrument semantic role information.

step	state	item
step1.2	step1.2_state1	device1
step1.2	step1.2_state2	device1
step1.2.1.1	step1.2.1.1_state1	boiling-waterBath
step1.2.1.1	step1.2.1.1_state2	boiling-waterBath
step1.2.2.1	step1.2.2.1_state1	microwaveOven
step1.2.2.1	step1.2.2.1_state2	microwaveOven
step3.2	step3.2_state1	electrophoresis
step3.2	step3.2_state_m3	electrophoresis
step3.2	step3.2_state_m1	electrophoresis
step3.2	step3.2_state2	electrophoresis
step3.2	step3.2_state_m4	electrophoresis
step3.2	step3.2_state_m4	glass_plate

**Fig. 4.** Result of Query1: Extract all devices involved in all steps of the Alkaline Agarose Gel Electrophoresis procedure

### SPARQL Query

Query1. Return all devices involved in a state of all steps (1.1, 1.2, 3)

```
SELECT ?step ?state ?item
WHERE { ?step rdf:type :Step.
?step :hasState ?state.
?state :involves ?item.
?item rdf:type :Device}
```

Figure 4 shows all of the instruments involved in any state for all steps of the Alkaline Agarose Gel Electrophoresis procedure. Using this information and knowledge about the steps in procedure, the instrument gel electrophoresis can be inferred.

## 6 Conclusions and Future Work

In this research we have provided prototypes for two ontologies of the biochemistry domain. The first ontology, *procedurally rhetorical frame semantics*, provides semantic roles for procedural verbs. The second ontology provides information about biochemical techniques. This ontology can be used to give information that does not appear in the scientific article text. To the best of our knowledge, no research has proposed or incorporated the idea of a semantic frame based on verb analysis to assist in the analysis of argumentation in biochemistry articles. Nor has any attempt been made to build an ontology of biochemical techniques and laboratory practices.

Our future goal is an in-depth argumentation analysis of biochemistry articles. Having access to the rhetorical moves that have been extracted using the two ontologies will enable a computationally feasible technique that will enable argumentation mining of more-detailed scientific knowledge than is currently available. This will be an important step towards providing researchers in Computational Argumentation working in domains with similar discourse structure with a means of using and evaluating the metrics we will develop. We have begun conducting an annotation study for both semantic roles [1] and rhetorical moves [2]. In addition, we have built a prototype ontology that we described in other work [3].

The SPARQL Query and Fig. 4 show the power of using the ontological knowledge to obtain relevant information about specific experimental processes<sup>7</sup>. We have also developed a set of frames for frequent procedural verbs (e.g., “digest”) in our analyzed data set. Our aim is to extend the VerbNet project by providing syntactic and semantic information for these procedural verbs. Further details can be found in the first author’s PhD thesis [4].

## References

1. **Alliheedi, M. & Mercer, R. E. (2019).** Semantic roles: Towards rhetorical moves in writing about experimental procedures. *Proceedings of the 32nd Canadian Conference on Artificial Intelligence*, pp. 518–524.
2. **Alliheedi, M., Mercer, R. E., & Cohen, R. (2019).** Annotation of rhetorical moves in biochemistry articles. *Proceedings of the 6th Workshop on Argument Mining*, pp. 113–123.
3. **Alliheedi, M., Wang, Y., & Mercer, R. E. (2019).** Biochemistry procedure-oriented ontology: A case study. *Proceedings of the 11th International Conference on Knowledge Engineering and Ontology Development*, To appear in Science and Technology Publications (SCITEPRESS), pp. .

<sup>7</sup>For additional detail about this prototype ontology, please see [3].

4. **Alliheedi, Mohammed (2019).** *Procedurally Rhetorical Verb-Centric Frame Semantics as a Knowledge Representation for Argumentation Analysis of Biochemistry Articles*. Ph.D. thesis, University of Waterloo.
5. **Baker, C. F., Fillmore, C. J., & Lowe, J. B. (1998).** The Berkeley FrameNet project. *Proceedings of the 17th International Conference on Computational Linguistics - Volume 1*, pp. 86–90.
6. **Biber, D. (1991).** *Variation Across Speech and Writing*. Cambridge University Press.
7. **Boyer, R. F. (2012).** *Biochemistry Laboratory: Modern Theory and Techniques*. Prentice Hall.
8. **Brown, N. G., Morrice, D. N., Beaud, G., Hardie, G., & Leader, D. P. (2000).** Identification of sites phosphorylated by the vaccinia virus B1R kinase in viral protein H5R. *BMC Biochemistry*, Vol. 1, No. 2.
9. **Carenbauer, A. L., Garrity, J. D., Periyannan, G., Yates, R. B., & Crowder, M. W. (2002).** Probing substrate binding to Metallo- $\beta$ -Lactamase L1 from *Stenotrophomonas maltophilia* by using site-directed mutagenesis. *BMC Biochemistry*, Vol. 3, No. 4.
10. **Chen, W. & Guidotti, G. (2001).** The metal coordination of sCD39 during ATP hydrolysis. *BMC Biochemistry*, Vol. 2, No. 9.
11. **Cohen, K. B. & Demner-Fushman, D. (2014).** *Biomedical Natural Language Processing*. Natural Language Processing 11. John Benjamins Publishing Company.
12. **Ester, C. & Uetz, P. (2008).** The FF domains of yeast U1 snRNP protein Prp40 mediate interactions with Luc7 and Snu71. *BMC Biochemistry*, Vol. 9, No. 29.
13. **Fillmore, C. J. (1976).** Frame semantics and the nature of language. *Annals of the New York Academy of Sciences*, Vol. 280, No. 1, pp. 20–32.
14. **Fillmore, C. J. (1977).** Topics in lexical semantics. In **Cole, R. W.**, editor, *Current Issues in Linguistic Theory*. Indiana University Press, pp. 76–138.
15. **Green, N. (2014).** Towards creation of a corpus for argumentation mining the biomedical genetics research literature. *Proceedings of the First Workshop on Argumentation Mining*, pp. 11–18.
16. **Green, N. (2015).** Identifying argumentation schemes in genetics research articles. *Proceedings of the 2nd Workshop on Argumentation Mining*, pp. 12–21.
17. **Huang, C.-C. & Lu, Z. (2016).** Community challenges in biomedical text mining over 10 years: Success, failure and the future. *Briefings in Bioinformatics*, Vol. 17, No. 1, pp. 132–144.
18. **Hunter, L. & Cohen, K. B. (2006).** Biomedical language processing: What's beyond PubMed? *Molecular Cell*, Vol. 21, No. 5, pp. 589–594.
19. **Hur, J., Özgür, A., Xiang, Z., & He, Y. (2012).** Identification of fever and vaccine-associated gene interaction networks using ontology-based literature mining. *Journal of Biomedical Semantics*, Vol. 3, No. 18.
20. **Kanoksilapatham, B. (2005).** *A corpus-based investigation of scientific research articles: Linking move analysis with multidimensional analysis*. Ph.D. thesis, Georgetown University.
21. **Krallinger, M., Leitner, F., Rodriguez-Penagos, C., & Valencia, A. (2008).** Overview of the protein-protein interaction annotation extraction task of BioCreative II. *Genome Biology*, Vol. 9, No. 2, pp. S4.
22. **Liakata, M., Saha, S., Dobnik, S., Batchelor, C., & Rebholz-Schuhmann, D. (2012).** Automatic recognition of conceptualization zones in scientific articles and two life science applications. *Bioinformatics*, Vol. 28, No. 7, pp. 991–1000.
23. **Lu, Z. (2011).** PubMed and beyond: A survey of web tools for searching biomedical literature. *Database*, Vol. 2011.
24. **Palmer, M., Gildea, D., & Kingsbury, P. (2005).** The Proposition Bank: An annotated corpus of semantic roles. *Computational Linguistics*, Vol. 31, No. 1, pp. 71–106.
25. **Ravikumar, K., Liu, H., Cohn, J. D., Wall, M. E., & Verspoor, K. (2012).** Literature mining of protein-residue associations with graph rules learned through distant supervision. *Journal of Biomedical Semantics*, Vol. 3, No. 3, pp. S2.
26. **Sambrook, J. & Russell, D. W. (2001).** *Molecular Cloning: A Laboratory Manual*. Cold Spring Harbor Laboratory Press.
27. **Schuler, K. K. (2005).** *VerbNet: A broad-coverage, comprehensive verb lexicon*. Ph.D. thesis, University of Pennsylvania.



28. **Segura-Bedmar, I., Martinez, P., & de Pablo-Sánchez, C. (2010).** Extracting drug-drug interactions from biomedical texts. *BMC Bioinformatics*, Vol. 11(Suppl5), No. P9.
29. **Swales, J. (1990).** *Genre Analysis: English in Academic and Research Settings*. Cambridge University Press.
30. **Thompson, P., Nawaz, R., McNaught, J., & Ananiadou, S. (2011).** Enriching a biomedical event corpus with meta-knowledge annotation. *BMC Bioinformatics*, Vol. 12, No. 393.

*Article received on 29/01/2019; accepted on 04/03/2019.  
Corresponding author is Mohammed Alliheedi.*



# Central Embeddings for Extractive Summarization Based on Similarity

Sandra J. Gutiérrez-Hinojosa, Hiram Calvo, Marco A. Moreno-Armendáriz

Instituto Politécnico Nacional, Centro de Investigación en Computación,  
Mexico City, Mexico

hcalvo@cic.ipn.mx

**Abstract.** In this work we propose using word embeddings combined with unsupervised methods such as clustering for the multi-document summarization task of DUC (Document Understanding Conference) 2002. We aim to find evidence that semantic information is kept in word embeddings and this representation is subject to be grouped based on their similarity, so that main ideas can be identified in sets of documents. We experiment with different clustering methods to extract candidates for the multi-document summarization task. Our experiments show that our method is able to find the prevalent ideas. ROUGE measures of our experiments are similar to the state of the art, despite the fact that not all the main ideas are found; as our method does not require annotated resources, it provides a domain and language independent way to create a summary.

**Keywords.** Extractive summarization, prevalent ideas extraction, concept similarity, central embeddings, DUC 2002.

## 1 Introduction

Automatic summarization is a challenging task, as there are many issues such as redundancy, temporal handling, co-reference, sentence order, etc. that need particular attention when summarizing multiple documents, thereby, making this task complex [6].

A summary contains the main ideas of documents; in order to perform this task automatically, there are two different approaches: paraphrasing the main ideas of a document, or extracting sentences from the documents representing main ideas. This work focuses on this latter approach, called extractive summarization. The purpose of

an algorithm for text summarization is to create a document formed by the most relevant information [4]; even for humans, this is a crucial step. There are several ways to determine which sentences are the most relevant in a set of documents.

Many algorithms to extract salient sentences from texts have been developed since the 1950s, when automatic text summarization arose. The first algorithm was based on topic representation, based on the idea that the more often a word repeats, the more likely it is to be important for identifying in the document [16]. This representation does not capture semantic and syntactic information; nevertheless, recent works with a similar approach have had a performance of 48% (recall) in a well-known dataset such as DUC (Document Understanding Conferences) 2002 [24]. Combining topic representation (word space models) with syntactic information such as Part Of Speech (POS) tagging helps to improve performance up to 55% (recall) [7].

In this work we propose using word embeddings combined with unsupervised clustering for the multi-document summarization task of DUC 2002. We aim to find evidence that semantic information is kept in word embeddings and this representation is subject to be grouped based on their similarity, so that main ideas can be identified in documents.

The following subsection describes related work to the task of document summarization, then in Section 2 we give some preliminaries related to this work. Our proposal is detailed in Section 3, Results are discussed in Section 4, and finally conclusions are drawn in Section 5.

### 1.1 Related Works

Word embedding is a distributed vector representation technique to capture information of a word. Each column of the vector represents a latent feature of the word and captures useful semantic properties [18]. This representation obtains good performance of 53% (recall) for the summarization task maximizing a submodular function defined by the sum of cosine similarities based on sentence embeddings [9] and 56% (recall) using an objective function defined by a cosine similarity based on document embeddings. This function is calculated based on the nearest neighbors distances on embedding distributions [10]. These works show that word embeddings are a useful representation to obtain the main ideas in the documents, but rely on the definition of an objective function adjusted to a particular domain.

The main stages to obtain an extractive summary are three: representation, scoring, and selection. The representation contains the relevant features of the text; in the scoring stage each sentence obtains a weight using a similarity metric and finally, in the selection stage the summary length constraint is satisfied.

In the original DUC 2002 competition, ten algorithms were submitted for the extractive summarization task (200 words length). [23] used weighted sentence scoring based on lexical content. This method scores sentences with higher values when the sentence is different from the others.

Some other techniques are topic-based, such as Latent Semantic Allocation (LSA), a probabilistic method that extracts semantic structure in the text that uses the document context for extracts information about word relations. A higher number of common words among sentences indicate that the sentences are semantically related. Another technique called Singular Value Decomposition (SVD) is a linear algebra method which finds the interrelations between sentences and words using matrix representation.

Wang et al. proposed a Bayesian sentence-based topic model by using the term-document and the term-sentences matrices; each row represents a term and each column represents the document

and the sentences, respectively. The goal of topic models is to infer words related to a topic and the topics discussed in a document. A higher value in each location indicates that the sentence or document is strongly related to that term [25].

The centroid-based method [20] is one of the most popular extractive summarization methods; it generates summaries using cluster centroids produced by topic detection, i.e. assesses the centrality of each sentence in a cluster and extracts the most important one.

A centroid is a set of words that are statistically important for the document cluster; therefore, the centroids are used to classify relevant documents and to identify salient sentences in a cluster. Each document is represented as a weighted vector of TF-IDF and the centroid is calculated using the first document. As new documents are processed, the TF-IDF values are compared with the centroid using cosine similarity, if the similarity is within a threshold, the new document is included in the cluster. The hypothesis of Radev et al. is that sentences containing words from the centroid are indicative of the topic of the cluster; the obtained results prove this hypothesis. However, the used word representation (TF-IDF) does not fully capture the semantic information of the words [20]

Formulating the summarization task as an optimization problem defines objective functions to evaluate candidate summaries. Objective functions are defined as essential parameters that a summary must accomplish, for example, coverage of all the main ideas [7]. Methods based on optimization methods have achieved best performance tested on DUC 2002. In Table 1 a summary of the best results is shown. A disadvantage of establishing objective functions is that they are adjusted based on a particular document set or domain, and thus, they might not represent a general way of creating summaries. This is why in this work we explore different ways of creating summaries, based on unsupervised clustering.

## 2 Preliminaries

In this section, details on the task in general (Section 2.1) are presented. Then we discuss

**Table 1.** Comparison of recall metrics for summaries

Work	Method	ROUGE-1	ROUGE-2	F-score	Dataset
Halteren (2002)	Scoring based on lexical content	0.2000	-	0.2100	DUC 2002
Radev et al. (2004a)	Centroid-cluster	0.4538	0.1918	-	Extracted by CDIR
Wang et al. (2009)	Position, semantic, LSA-NMF	0.4881	0.2457	-	DUC 2002
John et al. (2017)	Optimization	0.5532	0.2586	0.5419	DUC 2002

some text preprocessing techniques (Section 2.2), and evaluation methods (Section 2.3). The word embeddings used in this work are described in Section 2.4, along with the used similarity measures (Section 2.5).

## 2.1 The Summarization Task

The main goal of a summary is to encompass the main ideas in a document reducing the original document size. If all sentences in a text document were of equal importance, producing a summary would not be very effective, as any reduction in the size of a document would carry a proportional decrease in its informativeness. However, identifying the most relevant segments in the documents is the main challenge in summarization.

This task produces a transformation of source documents through content condensation by selecting and generalizing on important information [8]. Also, algorithms created for solving this task have a relevant application given the exponential growth of textual information online, and the need to find the main ideas of documents in a shorter time.

Research on the summarization task started to attract the attention of the scientific community in the late fifties when there was a particular interest in the automation of summarization for the production of abstracts of technical documentation [16].

### 2.1.1 Summarization Types

There are several distinctions in summarization, some are described below:

1. Source type: **single-document**, where a summary of a single document is produced; whereas in **multi-document** a summary of many documents on the same topic or the same event is built.
2. Output produced: **extractive**, which is a summary containing passages selected from the source document (usually sentences); and **abstractive**, where the information from the source document has been analyzed and transformed using paraphrasing, reorganizing, modifying and merging information for condensation.
3. Language: **mono-lingual**, where the language of the source document is the same for the summary; **multi-lingual**, which accepts two or more languages from a source document; and **cross-lingual**, which translates the summary to other than the original language.
4. Audience-oriented: **generic**, in this output it is assumed that anyone may end up reading the summary; and **query-oriented** that provides a summary that is relevant to a specific user query.

This work focuses on a multi-document, extractive, mono-lingual (English) and generic summary.

## 2.2 Text Preprocessing Techniques

Some of the most used techniques are:

- Word and sentence tokenization: Tokenization is the process of separating the text into words, phrases, symbols, or other meaningful elements called tokens. This process is considered easy compared to other tasks in natural language. However,

automatically extracted text may contain inaccurately compounded tokens, spelling errors and unexpected characters that can be propagated into later phases causing problems. Therefore, tokenization is an important step and in some cases needs to be customized to the data in question.

In all modern languages that are based on Latin, Cyrillic, or Greek classical languages, such as English, word tokens are delimited by a blank space. In these languages, also called segmented languages, token boundary identification is not a complex task, an algorithm which replaces white spaces with word boundaries and inserts a white space when a word is followed by a punctuation mark will produce a reasonable performance. Nonetheless, a period is an ambiguous punctuation mark indicating a full-stop, a part of abbreviation or both. Regular expressions can resolve these ambiguities defining different string search patterns [22]. In this work Python libraries have been used for tokenization<sup>1</sup>

- Stop words removal: The stop words are common and non-informative words that are often filtered, such as articles, prepositions, pronouns, etc. The removal of stop words have been done using methods based on Zipf's law [26], these methods indicate a distribution of words for any corpus and established an upper and lower cut-off frequency, being stop words the ones that are not between the cut-off.

Analyzing a dataset shows the most frequent words are document type dependent. A definitive stop words list does not exist, therefore the list used in this work is a general one<sup>2</sup> and contains 153 items.

- Stemming: This method is used to reduce words to a common form by removing their longest ending handling spelling exceptions. Two main principles are used in the construction of a stemming algorithm: iteration

and longest-match. Iteration is based on the fact that suffixes are attached to stems in a certain order, no more than one match is allowed within a single order-class; and the longest-match principle states that within any given class of endings, the longest ending should be removed.

The stemming algorithm<sup>3</sup> used in this work is based on [19].

## 2.3 Summary Evaluation

There are two quality evaluation methods for the summarization task: extrinsic and intrinsic. Extrinsic methods are based on the performance of a specific task (question-answering, comprehension, etc.) while intrinsic measures are based on norm set (fluency, coverage, similarity to an annotator summary, etc.).

Both quality evaluation methods can be performed by a human or a machine. The automatic evaluation lacks the linguistic skills and emotional perspective that a human has, but is popular because the evaluation process is quick, even when the summaries are large, and provides a consistent way of comparing the various summarization algorithms [3].

### 2.3.1 Recall-Oriented Understudy for Gisting Evaluation (ROUGE)

The University of Southern California's Information Sciences Institute (ISI) developed the recall-based metric called ROUGE-N defined by Equation 1 [15].

$$ROUGE-N = \frac{\sum_{s \in GSS} \sum_{n\text{-gram}} Count_{match}(n\text{-gram})}{\sum_{s \in GSS} \sum_{n\text{-gram}} Count(n\text{-gram})}, \quad (1)$$

where  $N$  is the number of  $n$ -grams,  $GSS$  is a set formed by the gold-standard summaries  $s$ ,  $Count_{match}(n\text{-gram})$  is the number of  $n$ -grams co-occurring in gold-standard and the retrieved summaries and  $Count(n\text{-gram})$  is the number of  $n$ -grams in the gold-standard summary [15].

In 2004, the ROUGE package was created including additional recall-based metrics, such as

<sup>1</sup>[https://www.nltk.org/\\_modules/nltk/tokenize.html](https://www.nltk.org/_modules/nltk/tokenize.html)

<sup>2</sup>based on [snowball.tartarus.org/algorithms/english/stop.txt](http://snowball.tartarus.org/algorithms/english/stop.txt)

<sup>3</sup>[http://www.nltk.org/\\_modules/nltk/stem/snowball.html](http://www.nltk.org/_modules/nltk/stem/snowball.html)

ROUGE-L, ROUGE-W, ROUGE-S, etc. and their precision and F-score metrics<sup>4</sup>. This package has a maximum reference count, i.e. if the word is repeated it only counts the number of times it is repeated in the gold-standard summary.

ROUGE metrics were evaluated to measure their correlation with human evaluations; for the multi-document summarization task, ROUGE-1 and ROUGE-2 showed high Pearson's correlation (90%) [14]. These two metrics are used in this work.

### 2.3.2 Document Understanding Conferences

In 2000, to foster progress in summarization, and as a part of an evaluation campaign organized by the Defense Advanced Research Projects Administration (DARPA) and the National Institute of Standards and Technology (NIST), the Document Understanding Conferences (DUC) were created<sup>5</sup>. In these challenges, different summarization tasks were developed by NIST and data for training and testing was distributed to participants.

In the dataset of DUC 2002 for the multi-document extractive summarization task (200 words length) 60 collections (document sets) with two gold-standard summaries each were distributed, but due to reasons beyond our knowledge, one document collection (d088) received no gold-standard summaries and two collections (d076 and d098) received only one; therefore, only 57 collections have the two gold-standard summaries. In this work, the dataset of DUC 2002 with 57 collections was used [23].

## 2.4 Word Embeddings

The term word embedding was originally coined by [1]. They trained a neural network to predict the next word given previous words in order to obtain a feature vector associated with each word; similar words are expected to have similar feature vectors. However, [2] were the first to demonstrate the power of pre-trained word embeddings and

establish word embeddings as a highly effective tool when used in natural language processing tasks. Moreover, in [18] word embeddings were brought to the fore through the creation of word2Vec and a tool-kit enabling the training and use of pre-trained embeddings.

Word2Vec is an efficient method for learning high-quality vector representations of words from large amounts of text data using neural networks. There are two models for computing embeddings: the bag-of-words and skip-gram models. Bag-of-words model predicts the probability of a word given a context, while the skip-gram model predicts the context given a word [17].

Word embeddings result from applying unsupervised learning, therefore they do not require annotated datasets. Rather, they can be derived from already available unannotated corpora.

### 2.4.1 Paragraph Embeddings

With the success of word embeddings, new algorithms called paragraph embeddings were developed. These paragraph embeddings, based on word embeddings, are an unsupervised learning algorithm that learns vector representations for variable length pieces of texts such as sentences and documents. As in word embeddings, there are two models: memory model and bag-of-words. Memory model predicts a paragraph identification given a number of context words, while the bag of words model ignores context words and forces the model to predict words randomly sampled from the paragraph in the output layer [12].

A software framework implementing these techniques was created [21] and the method was named doc2Vec<sup>6</sup>. In [11] they performed an empirical evaluation of doc2Vec on two tasks: duplicate question detection in a web forum and semantic textual similarity between two sentences, finding that doc2Vec in bag-of-words model performs better than the memory model. In this work, the final hyper-parameters and model of the previous work have been used<sup>7</sup>.

<sup>4</sup><http://rxnlp.com/rouge-2-0/>

<sup>5</sup><https://www-nlpir.nist.gov/projects/duc/intro.html>

<sup>6</sup><https://radimrehurek.com/gensim/models/doc2vec.html>

<sup>7</sup><https://github.com/jhlau/doc2vec>

## 2.5 Similarity Measure

Measuring similarity between vectors is related to measuring the distance between them, the smaller the distance the larger the similarity.

Finding similarity between words is a fundamental part of finding the sentence, paragraph and document similarities. Words can be similar in two ways: lexically and semantically. Words are similar lexically if they have a similar character sequence. Words are similar semantically if they are used in the same context [5].

Word and paragraph embeddings are a representation that contains lexical and semantic information in vector form, therefore to measuring their similarity vector distance has to be computed.

Cosine similarity measures the distance between two vectors using an inner product that measures the angle between them, as shown in Equation 2:

$$D = \frac{x \cdot y}{\|x\| \|y\|}. \quad (2)$$

Euclidean distance is the square root of the sum of squared differences between corresponding elements of the two vectors, also called L2-distance, as shown in Equation 3:

$$D = \sum_i^N \sqrt{(X_i - X_j)^2}, \quad (3)$$

where  $N$  is the dimension of each vector and  $i$  and  $j$  are the two vectors.

## 3 Proposal

The proposed method for the multi-document summarization task consists of three stages: (a) pre-process the dataset DUC 2002 in order to eliminate non-content words (Section 3.1) (b) select a word representation for this dataset to capture semantic and syntactic information and obtain sentence vectors (Section 3.2); and (c) implement a method to obtain the main ideas of the documents and select the relevant ones (Section 3.3). The sentences with main ideas will form the summary; this is the general approach to generate a summary, as described in Section 2.1. In Figure 1 a general diagram of the proposal is shown.

## 3.1 Pre-processing Stage

In the first stage sentence tokenization and word tokenization of each sentence are implemented using Python libraries based on regular expressions and named entities recognition<sup>8</sup>.

The DUC 2002 dataset has 547 documents  $D$  grouped in 57 document sets  $T$  with two gold-standard summaries each.

In this work, each set is tokenized in sentences and each sentence is tokenized in words; then the stop words have been removed using a list of non-informative words for the English language<sup>9</sup> and stemming of each word in a sentence has been done, based on Porter's stemming<sup>10</sup>, this is shown in Figure 2.

## 3.2 Embeddings Model

In the second stage, the word embedding model *doc2Vec* is used<sup>11</sup>. The model is a trained Artificial Neural Network (ANN) whose input is a set of tokenized words—which can be extracted from a sentence, a document, or a set of documents—and its output is a vector of 300 dimensions, called, from now on, simply *embedding*, i.e., an embedding is a vector representation of the sentence, document, or set of documents.

An example of the embedding model used in this work is shown in Figure 3, where the input is a sentence of the DUC dataset and the output is an embedding. The final hyper-parameters and model described in [11] have been used.

In order to find the sentence that has been transformed to a vector, let  $S_i^{j,t}$  be the  $i$ -th sentence belonging to the  $j$ -th document of the  $t$  set of documents, and  $E(S_i^{j,t})$  the embedding corresponding to  $S_i^{j,t}$ . Then we define a function  $f$  that associates each element in  $S$  set with an element in  $E(S)$ , as described by Equation 4:

$$f : E(S_i^{j,t}) \rightarrow S_i^{j,t}. \quad (4)$$

<sup>8</sup>[https://www.nltk.org/\\_modules/nltk/tokenize.html](https://www.nltk.org/_modules/nltk/tokenize.html)

<sup>9</sup><http://snowball.tartarus.org/algorithms/english/stop.txt>

<sup>10</sup>[http://www.nltk.org/\\_modules/nltk/stem/snowball.html#EnglishStemmer](http://www.nltk.org/_modules/nltk/stem/snowball.html#EnglishStemmer)

<sup>11</sup><https://github.com/jh1lau/doc2vec>



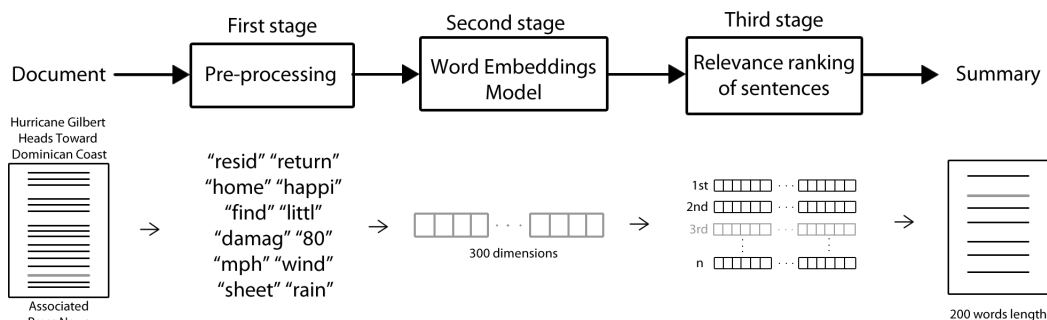


Fig. 1. General scheme for the proposal

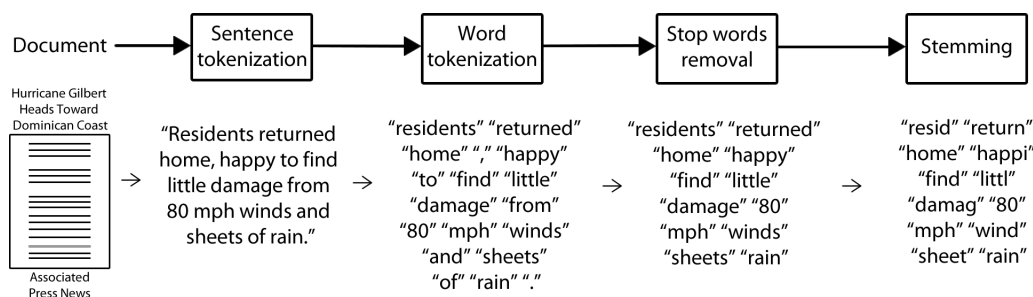


Fig. 2. Pre-processing stage

This function keeps an index to map a vector to the sentence that originated it (one to one function) in order to build the summary with the corresponding sentence from specific embeddings.

### 3.3 Relevance Ranking of the Sentences

In the last stage, once the embeddings for each sentence in the DUC dataset are obtained, we propose to calculate a central vector that contains the document main ideas, using the average of the sentence embeddings.

For this approach, we consider that each column of the sentence embedding represents a document subject and a higher value indicates the important subjects. Consequently, the average of the sentence embeddings in a set should represent the subjects in the set and higher value columns indicate the important subjects in the set; we call this average *central embedding* (CE) and consider that it represents the main idea of the document set.

The distance between each paragraph embedding and the central embedding in each column is short if both match in most columns; this means that the sentence embedding contains the same subjects than the main idea (central embedding) of the document set, and thus it is a relevant embedding that must be included in the summary; an example of this is shown in Figure 6.

In this toy example, four vectors with four columns each are shown. The average of each column is calculated, this vector is the central embedding and the cosine similarity with each sentence embedding is shown with the aim to show that the last sentence embedding has a higher similarity because the distance in each column is shorter than the other sentence embeddings.

Recalling that word embedding model input, shown in Figure 3, could be a word, a sentence, a paragraph, a document or text of any length. We propose three different forms of computing the central embedding: (a) using the sentence embeddings (**CE-S**), as in Figure 6; (b) using the document embeddings (**CE-D**), instead of using

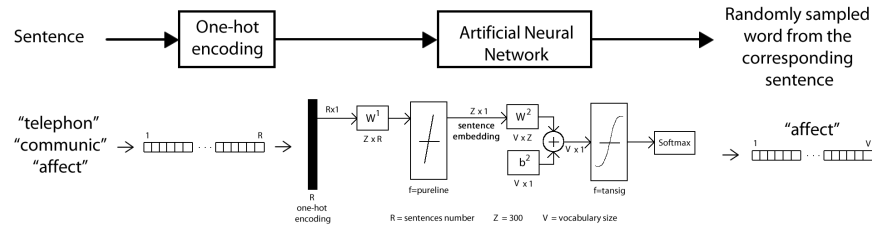


Fig. 3. The artificial neural network architecture used to train [11]

#### Algorithm 1 Central Embeddings calculation

```

1: procedure CE( $T^t$ )
2: input:  $T^t$ , a set of documents to summarize.
3: output:  $CES, CED, CESet$ , central embedding vectors per sentence, document and set.
4: Let  $D^{j,t}$  be the  $j$ -th document belonging to the  $t$  set of documents.
5: Let  $S_i^{j,t}$  be the  $i$ -th sentence belonging to the  $j$ -th document of the  $t$  set of documents.
6: Let  $E(S_i^{j,t})$  be the embedding corresponding to  $S_i^{j,t}$ .
7: Let  $E(D^{j,t})$  be the embedding corresponding to  $D^{j,t}$ .
8: Let  $E(T^t)$  be the embedding corresponding to all documents of the  $t$  set.
9:  $nsent_t \leftarrow |\bigcup_{i,j} S_i^{j,t}|$  ▷ the total number of sentences for a given set  $t$ 
10:  $ndoc_t \leftarrow |\bigcup_j D_j^{j,t}|$  ▷ the total number of documents for a given set  $t$ 
11:  $CES_t = \sum_{i,j} \frac{E(S_i^{j,t})}{nsent_t}$  ▷ Central Embedding per sentence
12:  $CED_t = \sum_j \frac{E(D_j^{j,t})}{ndoc_t}$  ▷ Central Embedding per document
13:  $CESet_t = E(T^t)$  ▷ Central Embedding per set
14: return  $CES, CED, CESet$ 
15: end procedure

```

the sentence embeddings in Figure 6 (i.e., the input in Figure 3 is a document); and (c) the central embedding is the document set embedding (**CE-Set**), this means that the input in Figure 3 is a document set. In Figure 4 these variants are illustrated. Algorithm 1 shows the pseudo-code for computing these three variants.

Once the central embedding is obtained, the cosine similarity between each sentence embedding and the calculated central embedding is calculated, giving ranked sentence embeddings related to a sentence using Equation 4. The top ranked sentences will form the summary. This process is depicted in Figure 5 and detailed in Algorithm 2. The central embedding  $CE$  can be CE-S, CE-D or CE-Set, as previously calculated.

## 4 Results

For the evaluation of the summaries in each experiment presented in Section 3, two measures have been used: ROUGE-1, ROUGE-2, and F-score because they have a high correlation with human evaluation of summaries [13].

Recalling that each document set contains two gold-standard summaries of 200 words length, we present the results comparing each gold-standard summary separately, and using both as an average result.

Three different forms of computing the central embedding were proposed: (a) using the sentences (**CE-S**), (b) using the documents (**CE-D**) and (c) using the document set (**CE-Set**). In Tables

**Algorithm 2** Sentence selection

---

```

1: procedure SELECT( $T^t, CE, length$ )
2: input:  $T^t$ , a set of documents to summarize,  $CE$  a central embedding,  $length$  (in words) of the summary
3: output: a summary of length  $length$ .
4:    $R_i^{j,t} \leftarrow sim_{cos}(E(S_i^{j,t}), CE) \quad \forall i, j$  ▷ cosine similarity
5:    $rSim \leftarrow rank_{top-down}(R_i^{j,t})$  ▷ create a list of embeddings ordered from most- to less-similar to  $CE$ 
6:    $nwords \leftarrow 0$  ▷ number of words
7:   for each  $e$  in  $rSim$  do ▷  $e$  is an embedding
8:     find  $S_i^{j,t}$  corresponding to  $e$  using eq. 4
9:     print  $S_i^{j,t}$ 
10:     $nwords \leftarrow nwords + |S_i^{j,t}|$ 
11:    if ( $nwords > length$ ) then
12:      return
13:    end if
14:  end for
15: end procedure

```

---

2, 3 and 4 ROUGE-1, ROUGE-2 and F-score for the experiments are shown.

**Table 2.** Recall for CE-S experiment

Gold-standard summary set	ROUGE-1	ROUGE-2	F-score
A	0.3808	0.1087	0.3740
B	0.3671	0.0932	0.3605
A and B	0.3740	0.1010	0.3673

**Table 3.** Recall for CE-D experiment

Gold-standard summary set	ROUGE-1	ROUGE-2	F-score
A	0.4371	0.1906	0.4495
B	0.4336	0.1873	0.4454
A and B	<b>0.4353</b>	<b>0.1889</b>	<b>0.4474</b>

**Table 4.** Recall for CE-Set experiment

Gold-standard summary set	ROUGE-1	ROUGE-2	F-score
A	0.4233	0.1791	0.4365
B	0.4119	0.1633	0.4239
A and B	0.4176	0.1712	0.4302

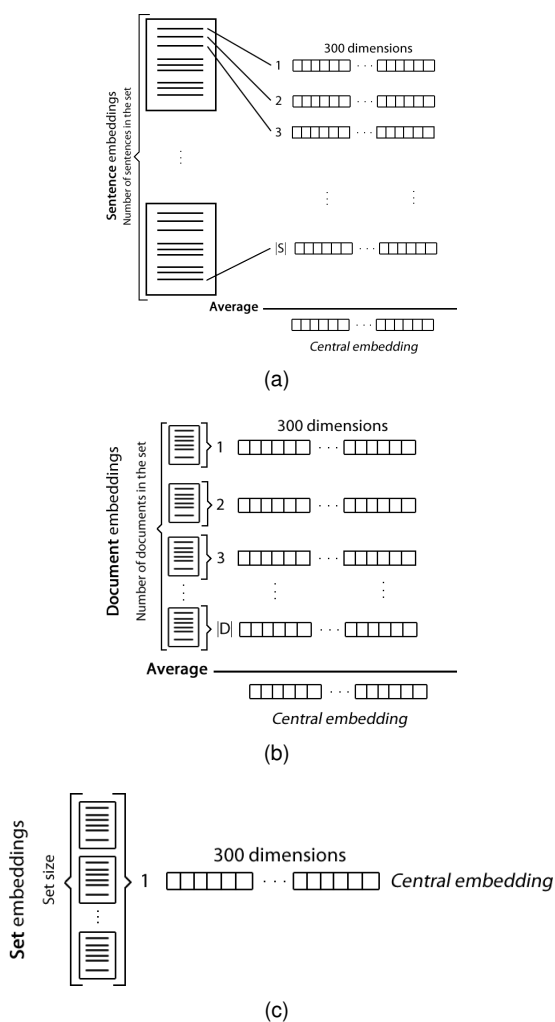
Our calculated embeddings for the DUC 2002 dataset, as well as the different Central Embeddings are available online<sup>12</sup>

<sup>12</sup><http://dx.doi.org/10.21227/qq4m-er38>

#### 4.1 Case Study of a Document Set

The main hypothesis of this proposal is that a central embedding should contain the main idea of a document and therefore, the sentence embeddings close to this central embedding contain important ideas because of their similarity. We will examine a particular document set, composed of 6 documents ( $|D^t| = 6$ ) and 480 sentences ( $|S^t| = 488$ ) dealing with the event of Hurricane Hilbert moving towards the coast. Different Associated Press reports from different countries (Jamaica, Santo Domingo, Mexico (Yucatan), USA (Miami)), and a Wall Street Journal article are included. Figure 7 shows the provided abstracts by two different annotators (Gold standard A and B). We have marked sentences according to the subject they cover. In yellow, sentences dealing with the hurricane's route are highlighted. Those reporting damages are marked in green, and orange highlights sentences dealing with hurricane's characteristic features. It can be seen that both summaries contain these three subjects in a balanced way.

Generated summaries of our system on this set of documents are shown in Figure 8. The first strategy (CE-S) includes several sentences (7, 10, 11, 12) that are not clearly identified under the three previously mentioned main subjects; few sentences are included on the damage report.



**Fig. 4.** Different ways of computing average vectors: (a) CE computed using the sentences (**CE-S**); (b) CE computed using the documents (**CE-D**) and; (c) CE using the document set (**CE-Set**).  $|S|$  is the number of sentences in each set, and  $|D|$  is the number of documents in each set

The second strategy (CE-D, Figure 8(b)) gives the best balance on the three subjects mentioned.

The third strategy (CE-Set, Figure 8(c)) lacks information on the route description. Although more descriptive sentences are obtained with the CE-S experiment, the best performance was obtained in the CE-D experiment because in CE-S

the sentence descriptions are not related with route and origin of the hurricane.

The effect of locating the central embedding with different strategies can be observed in Figure 9. These plots were created using the Radviz library<sup>13</sup>, which allows to project the 300-dimension embeddings into a 2D plot for visualization purposes. Circle markers indicate the document set sentences; triangle markers indicate the selected sentences; square markers indicate the location of the central embedding; cross markers indicate the central embeddings for the A gold standard, while star markers indicate the central embeddings for the B gold standard.

In Figure 9(a) the central embedding is located in the center of the selected sentences and very close to gold-standard central embeddings, but the sentences are short and do not contain relevant topics; this summary only selects one sentence from each gold-standard. In Figure 9(b) the central embedding seems far from the gold-standard central embeddings but contains more information from damage reports; it has two sentences in common with the gold-standard summaries.

In Figure 9(c) the central embedding is far from the gold-standard central embeddings, but is close to the central embedding of the previous experiment containing sentences with the damage report. It has five sentences in common with the previous experiment and two sentences in common with the gold-standard summaries.

## 5 Conclusions

In this paper we addressed the multi-document summarization task using a word embedding model that represents sentences as vectors which contain syntactic and semantic information.

Different variants to calculate central embeddings have been described. Specifically, three different ways of calculating averages were proposed: (1) using the sentences, (2) using the documents and (3) using the document set. Their foundations rely on the centroid-based method, which indicates that sentences containing

<sup>13</sup>[https://cran.r-project.org/web/packages/Radviz/vignettes/single\\_cell\\_projections.html](https://cran.r-project.org/web/packages/Radviz/vignettes/single_cell_projections.html)

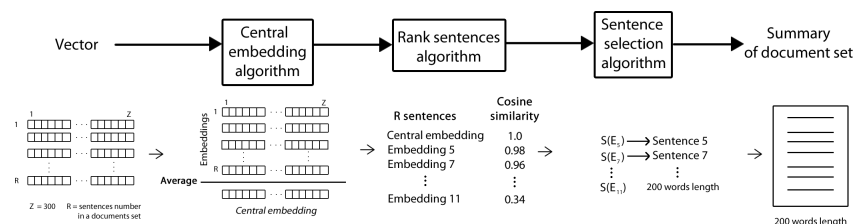


Fig. 5. Procedure to find relevant main ideas of documents set using Central Embeddings (CE)

					Cosine similarity
	-0.15	0.78	-0.95	0.35	0.27
	0.75	-0.82	0.75	0.57	0.44
	-0.18	-0.25	0.75	0.24	0.19
	0.21	0.72	-0.38	0.43	0.62
Average	0.15	0.10	0.04	0.39	
	Central embedding				

Fig. 6. Example of averaging vectors

words from the centroid are more indicative of the document topic. We found that using the documents for calculating the averages yielded better results when evaluated on the DUC 2002 corpus.

Our method obtains similar performance to LSA methods with the advantages that sentence embeddings do not have the curse of dimensionality of the matrices and are independent of the document type and language. This implies that sentence embeddings obtain semantic information useful for summaries and the results could be improved if different main ideas can be found in sentence groups, for example, sentences with a predominant description of the origin and route of the hurricane.

The advantages of this method are: it does not need any linguistic resource, it is easy to implement and has a similar performance to the state of the art. Also, our method is unsupervised, thus it can be adapted to other summarization corpora and language without the need of adjusting parameters, or estimating optimization goals.

In future work, we plan to evaluate the results with clustering algorithms in order to obtain

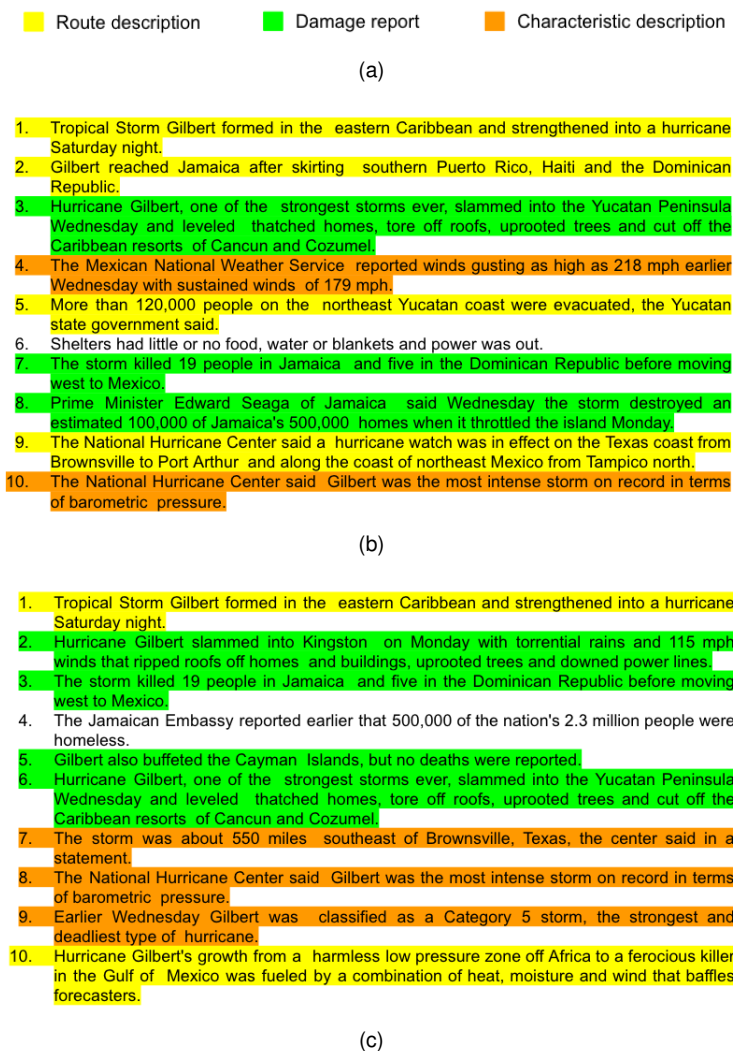
different groups of main ideas in order to capture the balance of topics observed in gold-standard summaries. Also, testing our method on different corpora to evaluate its performance is left as future work.

## Acknowledgments

The authors wish to thank the Government of Mexico (Instituto Politécnico Nacional, SNI, SIP-IPN, COFAA-IPN, BEIFI-IPN and CONACyT) for providing necessary support to carry out this research work.

## References

1. Bengio, Y., Ducharme, R., Vincent, P., & Jauvin, C. (2003). A neural probabilistic language model. *Journal of machine learning research*, Vol. 3, No. Feb, pp. 1137–1155.
2. Collobert, R. & Weston, J. (2008). A unified architecture for natural language processing: Deep neural networks with multitask learning. *Proceedings of the 25th international conference on Machine learning*, ACM, pp. 160–167.
3. Fiori, A. (2014). *Innovative document summarization techniques: Revolutionizing knowledge understanding*. IGI Global, Philadelphia.
4. Gambhir, M. & Gupta, V. (2017). Recent automatic text summarization techniques: a survey. *Artificial Intelligence Review*, , No. 1, pp. 1–66.
5. Gomaa, W. H. & Fahmy, A. A. (2013). A survey of text similarity approaches. *International Journal of Computer Applications*, Vol. 68, No. 13.
6. Gupta, V. & Lehal, G. S. (2010). A survey of text summarization extractive techniques. *Journal of emerging technologies in web intelligence*, Vol. 2, No. 3, pp. 258–268.



**Fig. 7.** Gold standard summaries from DUC 2002 (document set d061j) (a) legend; (b) Gold Standard A and; (c) Gold Standard B

7. John, A., Premjith, P., & Wilscy, M. (2017). Extractive multi-document summarization using population-based multicriteria optimization. *Expert Systems with Applications*, Vol. 86, pp. 385–397.
8. Jones, K. S. (2007). Automatic summarising: The state of the art. *Information Processing & Management*, Vol. 43, No. 6, pp. 1449–1481.
9. Kågebäck, M., Mogren, O., Tahmasebi, N., & Dubhashi, D. (2014). Extractive summarization using continuous vector space models. *Proceedings of the 2nd Workshop on Continuous Vector Space Models and their Compositionality (CVSC)@ EACL*, pp. 31–39.
10. Kobayashi, H., Noguchi, M., & Yatsuka, T. (2015). Summarization based on embedding distributions. *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing*, pp. 1984–1989.
11. Lau, J. H. & Baldwin, T. (2016). An empirical evaluation of doc2vec with practical insights into document embedding generation. *arXiv preprint arXiv:1607.05368*.

1. Officials in the Dominican Republic, sideswiped Sunday by the storm, reported five dead.
2. Hurricane Gilbert Heading for Jamaica With 100 MPH Winds
3. The storm killed 19 people in Jamaica and five in the Dominican Republic before moving west to Mexico.
4. San Juan, on the north coast, had heavy rains and gusts Saturday, but they subsided during the night.
5. The National Hurricane Center said Gilbert was the most intense storm on record in terms of barometric pressure.
6. Hurricane Gilbert Heads Toward Dominican Coast
7. Airports in the region were closed.
8. "It's moving at about 17 mph to the west and normally hurricanes take a northward turn after they pass central Cuba".
9. Forecasters said the hurricane's track would take it about 50 miles south of southwestern Haiti.
10. Warnings were discontinued for the Dominican Republic.
11. "This time of year in the northwest Caribbean is best for development," Clark said.
12. What Makes Gilbert So Strong?
13. Gilbert Reaches Jamaican Capital With 110 Mph Winds
14. The storm was about 550 miles southeast of Brownsville, Texas, the center said in a statement.
15. The National Weather Service in San Juan, Puerto Rico, said Gilbert was moving westward at 15 mph with a "broad area of cloudiness and heavy weather" rotating around the center of the storm

(a)

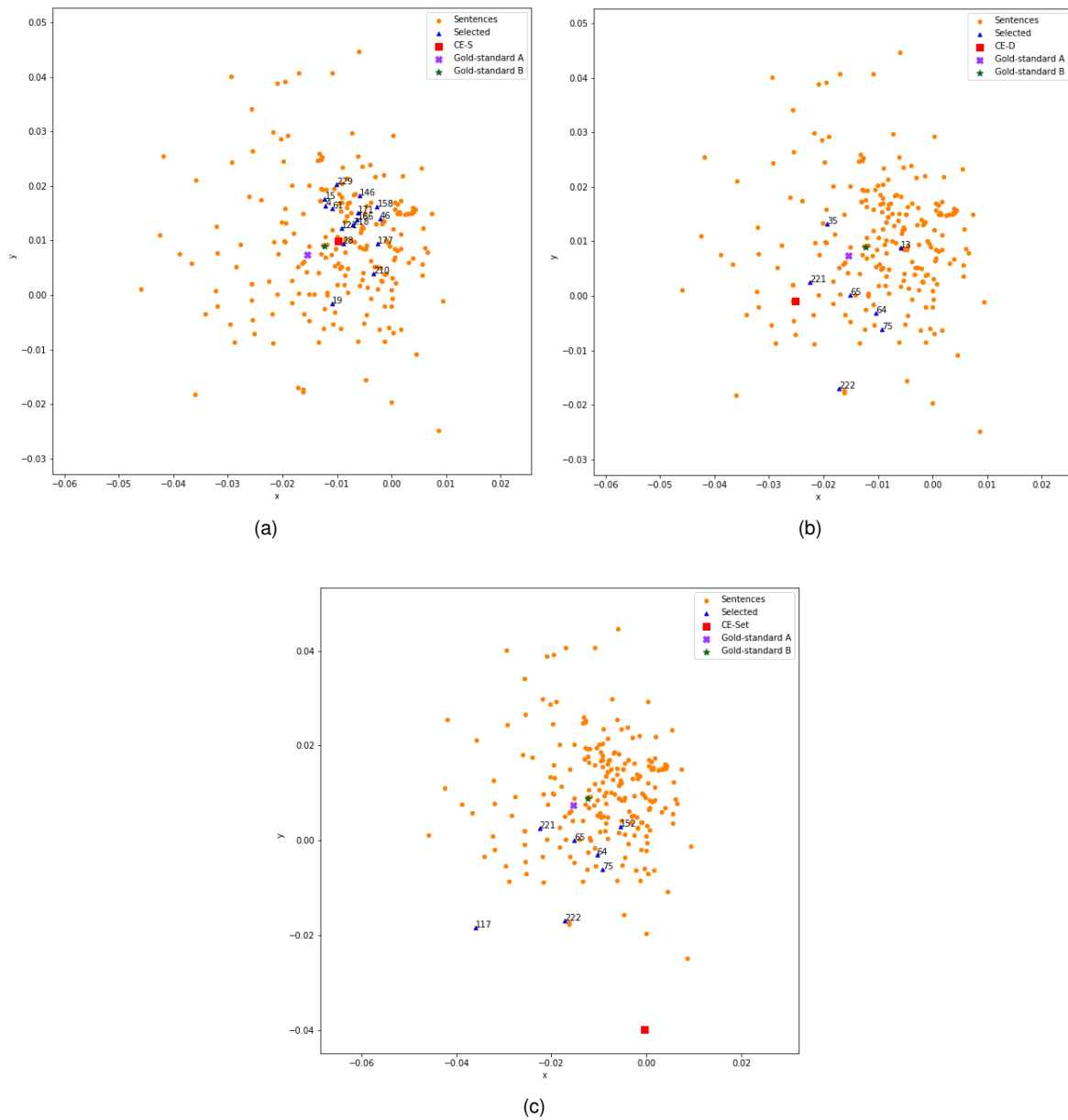
1. Hurricane Gilbert swept toward Jamaica yesterday with 100-mile-an-hour winds, and officials issued warnings to residents on the southern coasts of the Dominican Republic, Haiti and Cuba.
2. The storm ripped the roofs off houses and flooded coastal areas of southwestern Puerto Rico after reaching hurricane strength off the island's southeast Saturday night.
3. Hurricane Gilbert slammed into Kingston on Monday with torrential rains and 115 mph winds that ripped roofs off homes and buildings, uprooted trees and downed power lines.
4. The storm ripped the roofs off houses and caused coastal flooding in Puerto Rico.
5. Heavy rain and stiff winds downed power lines and caused flooding in the Dominican Republic on Sunday night as the hurricane's center passed just south of the Barahona peninsula, then less than 100 miles from neighboring Haiti.
6. Tropical Storm Gilbert formed in the eastern Caribbean and strengthened into a hurricane Saturday night.
7. Hurricane Gilbert, packing 110 mph winds and torrential rain, moved over this capital city today after skirting Puerto Rico, Haiti and the Dominican Republic.

(b)

1. Hurricane Gilbert swept toward Jamaica yesterday with 100-mile-an-hour winds, and officials issued warnings to residents on the southern coasts of the Dominican Republic, Haiti and Cuba.
2. Hurricane Gilbert slammed into Kingston on Monday with torrential rains and 115 mph winds that ripped roofs off homes and buildings, uprooted trees and downed power lines.
3. Hurricane Gilbert, one of the strongest storms ever, slammed into the Yucatan Peninsula Wednesday and leveled thatched homes, tore off roofs, uprooted trees and cut off the Caribbean resorts of Cancun and Cozumel.
4. The storm ripped the roofs off houses and flooded coastal areas of southwestern Puerto Rico after reaching hurricane strength off the island's southeast Saturday night.
5. The storm ripped the roofs off houses and caused coastal flooding in Puerto Rico.
6. Heavy rain and stiff winds downed power lines and caused flooding in the Dominican Republic on Sunday night as the hurricane's center passed just south of the Barahona peninsula, then less than 100 miles from neighboring Haiti.
7. As Gilbert moved away from the Yucatan Peninsula Wednesday night, the hurricane formed a double eye, two concentric circles of thunderstorms often characteristic of a strong storm that has crossed land and is moving over the water again.

(c)

Fig. 8. Generated summaries (for DUC 2002 document set d061j) (a) CE-S; (b) CE-D; (c) CE-Set



**Fig. 9.** Visualization of the different forms of computing central embeddings for the d061j document set: (a) CE calculated using the sentences (**CE-S**); (b) CE calculated using the documents (**CE-D**) and; (c) CE using the document set (**CE-Set**)



12. **Le, Q. & Mikolov, T. (2014).** Distributed representations of sentences and documents. *International Conference on Machine Learning*, pp. 1188–1196.
13. **Lin, C. Y. (2004).** Looking for a few good metrics: Automatic summarization evaluation - how many samples are enough? *Proceedings of the NTCIR Workshop 4*.
14. **Lin, C.-Y. (2004).** Rouge: A package for automatic evaluation of summaries. *Text Summarization Branches Out*.
15. **Lin, C.-Y. & Hovy, E. (2003).** Automatic evaluation of summaries using n-gram co-occurrence statistics. *Proceedings of the 2003 Conference of the North American Chapter of the Association for Computational Linguistics on Human Language Technology-Volume 1*, Association for Computational Linguistics, pp. 71–78.
16. **Luhn, H. P. (1958).** The automatic creation of literature abstracts. *IBM Journal of research and development*, Vol. 2, No. 2, pp. 159–165.
17. **Mikolov, T., Chen, K., Corrado, G., & Dean, J. (2013).** Efficient estimation of word representations in vector space. *arXiv preprint arXiv:1301.3781*.
18. **Mikolov, T., Sutskever, I., Chen, K., Corrado, G., & Dean, J. (2013).** Distributed representations of words and phrases and their compositionality. *Proceedings of the 26th International Conference on Neural Information Processing Systems - Volume 2, NIPS'13*, Curran Associates Inc., USA, pp. 3111–3119.
19. **Porter, M. F. (2001).** Snowball: A language for stemming algorithms.
20. **Radev, D. R., Jing, H., Styś, M., & Tam, D. (2004).** Centroid-based summarization of multiple documents. *Information Processing & Management*, Vol. 40, No. 6, pp. 919–938.
21. **Rehurek, R. & Sojka, P. (2010).** Software framework for topic modelling with large corpora. *In Proceedings of the LREC 2010 Workshop on New Challenges for NLP Frameworks*, Citeseer.
22. **Thompson, K. (1968).** Programming techniques: Regular expression search algorithm. *Communications of the ACM*, Vol. 11, No. 6, pp. 419–422.
23. **van Halteren, H. (2002).** Writing style recognition and sentence extraction.
24. **Wang, D. & Li, T. (2012).** Weighted consensus multi-document summarization. *Information Processing & Management*, Vol. 48, No. 3, pp. 513–523.
25. **Wang, D., Zhu, S., Li, T., & Gong, Y. (2009).** Multi-document summarization using sentence-based topic models. *Proceedings of the ACL-IJCNLP 2009 Conference Short Papers*, Association for Computational Linguistics, pp. 297–300.
26. **Zipf, G. K. (1949).** Human behaviour and the principle of least-effort. cambridge ma edn. *Reading: Addison-Wesley*.

Article received on 16/01/2019; accepted on 20/03/2019.  
Corresponding author is Hiram Calvo.



# Predicting and Integrating Expected Answer Types into a Simple Recurrent Neural Network Model for Answer Sentence Selection

Sanjay Kamath<sup>1,3</sup>, Brigitte Grau<sup>1,2</sup>, Yue Ma<sup>3</sup>

<sup>1</sup> LIMSI, CNRS, Université Paris-Saclay, Orsay, France

<sup>2</sup> ENSIIE, Université Paris-Saclay, Évry, France

<sup>3</sup> LRI, Univ. Paris-Sud, CNRS, Université Paris-Saclay, Orsay, France

{sanjay, bg}@limsi.fr, yue.ma@lri.fr

**Abstract.** Since end-to-end deep learning models have started to replace traditional pipeline architectures of question answering systems, features such as expected answer types which are based on the question semantics are seldom used explicitly in the models. In this paper, we propose a convolution neural network model to predict these answer types based on question words and a recurrent neural network model to find sentence similarity scores between question and answer sentences. The proposed model outperforms the current state of the art results on an answer sentence selection task in open domain question answering by 1.88% on MAP and 2.96% on MRR scores.

**Keywords.** Question answering, deep learning, answer sentence selection, expected answer types, sentence similarity.

## 1 Introduction

Question answering systems in recent times have mainly been dominated by neural network approaches that fetch state of the art results across different NLP tasks. Open domain question answering tasks include answer sentence selection, reading comprehension, multi-hop reasoning and reading etc. An example of a question answer pair from a dataset:

*Q: How a water pump works?*

*A: pumps operate by some mechanism ... to perform mechanical work by moving the fluid.*

An answer sentence selection model would retrieve the entire sentence from a paragraph as an answer. A common goal of the neural network models is to build end to end approaches which do not rely on intermediate tools or data provided by other systems. Some recent works such as BERT [3] and ELMO [11] use pre-trained language models trained with large neural network architectures and use it to fine tune downstream NLP tasks. These methods outperform current state of the art systems for reading comprehension as well as many other tasks. However, training such models on large datasets and the requirement of large scale computation power is sometimes not a feasible solution.

Other state of the art models such as QANet [19] on SQUAD and other end to end approaches try to implicitly learn information such as entity types, part of the speech tags, named entities, syntactic dependencies etc. and perform downstream tasks. But the challenge still remains in understanding whether or not they utilize such information implicitly or just overfit over the datasets and their unintended bias. A feasible yet challenging approach would be to utilize both the power of neural networks approaches and explicit

information such as entity types, dependencies, tags, together. Expected Answer Types (referred to as EAT hereafter) is one such vital information which is important for question answering systems to detect which type of answers do the questions require. Some examples of EAT with questions are listed below:

*Question: Which NFL team represented the AFC at Super Bowl 50?*

*Expected Answer Type: HUM.*

*Question: Where was franz kafka born ?*

*Expected Answer Type: LOC.*

[15] refer to this information as *Question Classes* in their work and show a significant improvement over a previous state of the art DNN model on TrecQA dataset which uses only word level information.

Our contributions in this article are as follows. We introduce two different ways of using *Question Classes* which is further referred as *EAT or Expected Answer Types* and experiment with several datasets along with TrecQA to determine if this would work better for a wider range of large scale datasets by using a simple model of a recurrent neural network which uses a pre-attention mechanism. To annotate other datasets apart from TrecQA, with EAT information, we propose a multiclass classifier model which is trained on a dataset built by using an existing rule-based system which predicts EAT for questions.

We report our findings on WikiQA, SQUAD-Sent and TrecQA dataset performance and show that we outperform state of the art results on TrecQA dataset<sup>1</sup> by the two different ways of highlighting Expected Answer Types in the data.

Answer sentence selection task has been extensively studied with different approaches ranging from n-gram models to neural network models. In former feature based QA systems, the Expected Answer Type (EAT) has been shown as a very important feature [7].

<sup>1</sup>[https://aclweb.org/aclwiki/Question\\_Answering\\_\(State\\_of\\_the\\_art\)](https://aclweb.org/aclwiki/Question_Answering_(State_of_the_art))

The EAT corresponds to an entity type organized in answer type taxonomies, as in [8] for the open domain or semantic types in biomedical domain as in [5].

Recent works on this task focus mainly on convolutional neural network approaches. [14] propose a CNN model using learning to rank approach, which computes a representation of both entries, candidate passage and question, and a similarity between these two representations using a pooling layer followed by similarity matrix computation. In [18], the similarity of the two entries is evaluated by computing interactions between words of the two texts by an attention layer. [4] propose a Multi-Perspective CNN for this task which was further used by [13] with a triplet ranking loss function to learn pairwise ranking from both positive and negative samples.

[15] use the same model but use Question Classes to enhance the dataset with highlighting entities in it. Highlighting entities were done by mainly two ways called Bracketing (appending a special token before and after the entity occurrence) and Replacement (replacing the entity word with a special token) methods. Our work uses a similar technique by replacing the entity word with special tokens but allows to learn them according to the expected types. The leaderboard of TrecQA evaluation<sup>1</sup> reports the state of the art scores from different methods reported by several articles.

## 2 Answer Sentence Selection

Answer sentence selection is a question answering task which is also referred sometimes as sentence reranking task. The task involves reranking a set of sentences  $S = \{S_1, \dots, S_m\}$  for a question  $Q$ , so that the correct sentences are ranked first. Sentence set  $S$  can contain the mixture of both negative and positive sentences relevant to the question, often more than one positive sentence.

We model this task as a pairwise similarity scoring task. For each sentence related to a question, we compute a similarity score against the question sentence and answer sentence. i.e.,  $(Q_i - S_{i,j}, Q_i - S_{i,j+1}, Q_i - S_{i,j+2}, \dots, Q_i - S_{i,j+n})$ .

## 2.1 RNN-Similarity

Recurrent neural networks such as LSTMs and GRUs are widely used in several NLP tasks like machine translation, sequence tagging, and question answering tasks such as reading comprehension and answer sentence selection.

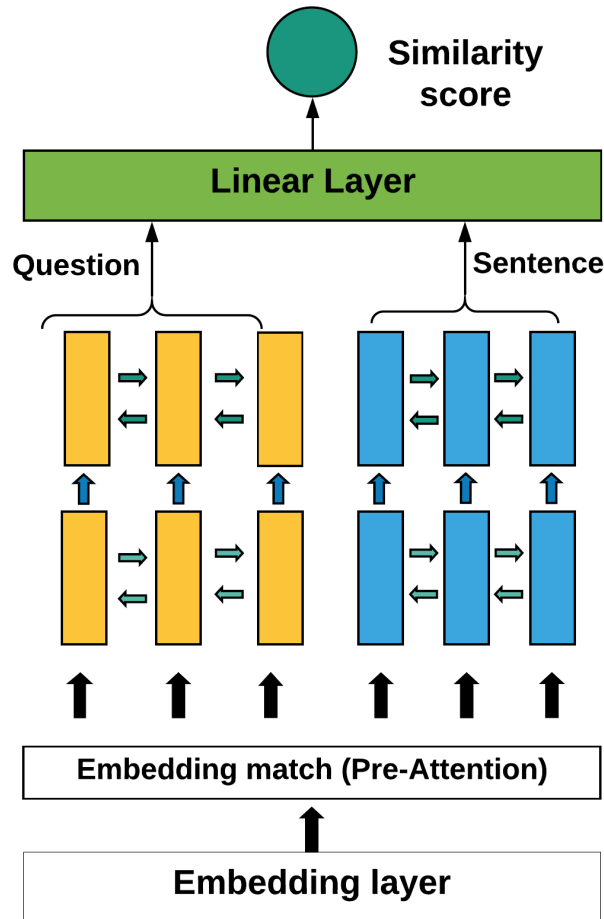


Fig. 1. Proposed RNN-Similarity model

We propose a simple model with recurrent neural networks and an attention mechanism to capture sequential semantic information of words in both questions and sentences and predict similarity scores between them. We refer to this model further in this article as *RNN-Similarity* model. Figure 1 shows the architecture of the model.

Question words  $Q = \{q_1, \dots, q_m\}$  and Sentence words  $S = \{s_1, \dots, s_n\}$  are sequences which are encoded using an embedding layer of dimension  $D$ :

$$E(Q) = \{E(q_1), \dots, E(q_m)\}, \quad (1)$$

$$E(S) = \{E(s_1), \dots, E(s_n)\}, \quad (2)$$

A pre-attention mechanism captures the similarity between sentence words and questions words in the same layer. For this purpose, a feature  $\mathcal{F}_{align}$  shown in Equation 3 is added as a feature to the LSTM layer:

$$\mathcal{F}_{align}(p_i) = \sum_j a_{i,j} E(q_j), \quad (3)$$

where  $a_{i,j}$  is,

$$a_{i,j} = \frac{\exp(\alpha(E(s_i)) \cdot \alpha(E(q_j)))}{\sum_{j'} \exp(\alpha(E(s_i)) \cdot \alpha(E(q_{j'})))}, \quad (4)$$

which computes the dot products between nonlinear mappings of word embeddings of question and sentence.

The above process is similar to [1] who use LSTMs to model Question and Paragraph to encode the words for reading comprehension task. We use 3-layer Bidirectional LSTM layers for both question and sentence encodings:

$$\{E(q_1), \dots, E(q_n)\} = \text{Bi-LSTM}(\{\tilde{E}(q_1), \dots, \tilde{E}(q_n)\}), \quad (5)$$

$$\{E(s_1), \dots, E(s_n)\} = \text{Bi-LSTM}(\{\tilde{E}(s_1), \dots, \tilde{E}(s_n)\}). \quad (6)$$

The LSTM output states are further connected to a linear layer and a sigmoid non-linear activation function is applied on the output of the linear layer which outputs the score ranging between 0-1, which signifies the similarity between the question and the answer sentence.

For the Expected Answer Types (EAT) version of question and sentences, we create special tokens for the entity type that are used for encoding the question  $Q$  and each sentence  $S$ .

**Table 1.** Three methods for replacing entities along with an example from TrecQA dataset

-	Method	Question	Sentence
1	Original text	Who is the author of the book, 'The Iron Lady: a biography of Margaret Thatcher'	in 'The Iron Lady,' <i>Young</i> traces ..... the greatest woman political leader since <i>Catherine the Great</i> .
2	Replacement - [15] (EAT Single type)	Who is the author of the book, 'The Iron Lady: a biography of Margaret Thatcher' <i>max_entity_left</i> <i>entity_left</i>	in 'The Iron Lady,' <i>max_entity_left</i> traces ..... the greatest woman political leader since <i>entity_left</i> .
3	EAT (Different types)	Who is the author of the book, 'The Iron Lady: a biography of Margaret Thatcher' <i>max_entity_left</i> <i>entity_hum</i>	in 'The Iron Lady,' <i>max_entity_left</i> traces ..... the greatest woman political leader since <i>entity_hum</i> .
4	EAT (MAX + Different types)	Who is the author of the book, 'The Iron Lady: a biography of Margaret Thatcher' <i>max_entity_hum</i> <i>entity_hum</i>	in 'The Iron Lady,' <i>max_entity_hum</i> traces ..... the greatest woman political leader since <i>entity_hum</i> .

## 2.2 Highlighting Single Entity and Multiple Entity Types

The authors of [15] propose a method of replacing words by special token embeddings for highlighting entities that catch the EAT entity in sentences. In our work, this method is referred to as "EAT (single type)" in the following experiments. The entities belong to (HUM, LOC, ABBR, DESC, NUM or ENTY). HUM refers to a description, group, individual, title. LOC refers to city, country, mountain, state. ABBR refers to abbreviation, expansion. DESC refers to a definition, description, manner, reason. NUM refers to numerical values such as code, count, date, distance, money, order etc. ENTY refers to a numerous entity types such as animal, body, color, creation, currency, disease etc. More details regarding the taxonomy can be found in [9].

The entities, irrespective of which class they belong to, are treated similarly by replacing them by two special tokens *entity\_left* for entity

occurrences and *max\_entity\_left* for maximum occurring entity that corresponds to an entity that is at least twice the number of occurrences when compared to the second maximum occurring entity. Entity types are recognized using the named entity recognition tool. When an entity type in a sentence matches the EAT from the question, *entity\_left* token is used to replace the entity mentions in the sentences; same applies for the maximum occurring entity token *max\_entity\_left* as well.

Our proposition is to replace an entity according to the type it belongs to instead of replacing all kinds of entity by just one word i.e. *entity\_left*. We do it based on the different types of EAT it belongs to based on the taxonomy used in the original work. The intuition behind this method is that the model would learn to better map the relations between question words and specific entity type tokens when used in a model with attention mechanisms, rather than learning the relation between question words and the same generic entity type token for all entities.

This way, we can learn a different behaviour with an entity about location and with an entity about a person for example.

The example in Table 1 line 3 refers to an example that has an EAT as “HUM” from the taxonomy, so we replace it as *entity\_hum*. We do the same for other expected answer types such as *entity\_loc* for “LOC” type, *entity\_enty* for “ENTY” type, *entity\_num* for “NUM” type, *entity\_desc* for “DESC” type, *entity\_abbr* for “ABBR” type. We replace the entity mentions in the text whose types are matching the EAT from questions.

We also experiment with a variant where the *max\_entity\_left* is replaced with the entity type along with other entities. If the maximum entity is of type “HUM”, we replace it as *max\_entity\_hum*. This method is referred to as “EAT (MAX + different types)” in the following experiments. We create a random word embedding ranging between (-0.5 - 0.5) with dimension  $D$  for each of the EAT words and encode the word with this embedding when it appears in all our experiments.

### 3 Experiments

We perform experiments on three datasets namely 1) TrecQA, 2) WikiQA, 3) SQUAD-Sent with and without EAT annotations. Thus we had to develop our own annotation tools.

#### 3.1 Annotation of the EAT

Since SQUAD-EAT (see section 3.3) is the result of a rule-based method with a high accuracy score (97.2% as reported in [9]), we use it to train a multiclass classifier based on a CNN model for text classification<sup>2</sup> by [6], by modifying the outputs into a multi-class setting. We further refer to this as EAT Classifier. We use 300 dimensions GloVe embeddings by [10].

The output classes of the classifier refer to a type based on the taxonomy such as *ABBR*, *DESC*, *ENTY*, *HUM*, *LOC*, *NUM* and a “NO.EAT” class to signify an EAT which is not in the above list of classes. We do not use the fine level taxonomy in this work because of a resulting large number of

classes with sparse distribution of samples in the dataset. Below is an example from SQUAD-EAT with HUM:

**Question:** Which NFL team represented the AFC at Super Bowl 50?

**Expected Answer Type:** HUM.

We train the multi-class classifier model using the SQUAD-EAT dataset which gets an accuracy score of 95.17% on the SQUAD-EAT dev in our experiment, according to the annotation done by [15] as reference.

#### 3.2 Annotation of the Entities

We detect the entities in the sentences using Dbpedia Spotlight tool by [2]. The detected entities by spotlight are verified for their entity type match using the Spacy NER tool which is mapped to EAT using the mapping shown in table 2. Only the matching entities are highlighted and others are discarded. We replace the special token by adding one for the maximum occurring entity, which is described in Section 2.2.

#### 3.3 The Data

TrecQA dataset is a standard dataset used to benchmark state of the art systems for answer sentence selection task. The authors of [9, 15] provide the EAT annotations for the TrecQA dataset based on their rule-based approach.

We modify the QA dataset SQUAD by [12] designed for machine comprehension, into an answer sentence selection dataset to provide the answers in their original context. We name it as *SQUAD-Sent*. We do this by processing the dataset where each example is usually a triple of Question, Paragraph and Answer span (Text and the answer start offset in the paragraph) into a dataset where each triple is a Question, Sentence and Sentence label. The sentence label is 1 if the answer is present inside the sentence, else it is 0. We perform sentence tokenization using spacy toolkit<sup>3</sup> on paragraphs of SQUAD and perform a check for an exact match of answer strings in them.

*SQUAD-Sent* is a special case dataset where there is just one positive sentence per question and

<sup>2</sup><https://github.com/cmasch/cnn-text-classification>

<sup>3</sup><https://spacy.io>

**Table 2.** Spacy named entity annotation scheme following OntoNotes 5 corpus mapped with EAT types

EAT	Spacy annotated tag
HUM	PERSON, ORG, NORP
LOC	LOC, GPE
ENTY	PRODUCT, EVENT, LANGUAGE, WORK_OF_ART, LAW, FAC
NUM	DATE, TIME, PERCENT, MONEY, QUANTITY, ORDINAL, CARDINAL

the other sentences are negative examples. The motivation to do this is because of the large scale property of this dataset, compared to the other datasets, with human-generated questions. For the expected answer types of SQUAD questions, we use SQUAD-EAT which is a dataset with EAT annotated questions on SQUAD v1 dataset questions which is annotated by the authors of [9, 15] on our request. WikiQA dataset by [17] is another dataset used for answer sentence selection task which was built using Bing search engine query logs. We use a preprocessed version as used by [13] which has removed certain examples without any positive answers and questions with more than 40 tokens to compare the scores. The questions and answer sentences are annotated with EAT information as described in section 3.1.

Table 3 shows the statistics of the datasets with EAT annotated questions and plain word level questions (regular datasets) and the number of entities annotated in each set. EAT version of TrecQA dataset is as reported in [15] and available through this link<sup>4</sup>.

### 3.4 Implementation

We implement the RNN-Similarity model in Pytorch, and we use MSELoss (Mean Squared Error loss) to minimize the error of predictions for relevance scores. We use adamax optimizer and keep the missing word embeddings as zero embeddings. We implement the EAT Classifier using the CNN model available online<sup>5</sup> and used Keras to implement the multiclass classifier which uses GloVe embeddings as input. The

<sup>4</sup>[www.harishmadabushi.com/research/answer-selection/](http://www.harishmadabushi.com/research/answer-selection/)

<sup>5</sup>[https://github.com/yoonykim/CNN\\_sentence](https://github.com/yoonykim/CNN_sentence)

**Table 3.** Statistics of datasets with plain and EAT annotated questions. '#' refers to "Number of."

Dataset	Split	#Plain Q	#EAT Q	#Entities
Trec QA	Train	1229	649	9064
	Dev	82	76	382
	Test	100	82	597
SQUAD-Sent	Train	87,599	78,740	35087
	Dev	10,570	9,606	4757
	Test	-	-	-
Wiki QA	Train	873	859	132
	Dev	126	124	4
	Test	243	236	38

code for both the models along with default hyperparameters is publicly available on Github <sup>6</sup>.

### 3.5 Results

Table 4 shows various results on different versions of datasets. Note that the questions in the following experiments of Table 4 contain all the questions from the datasets, which includes questions which are highlighted with EAT and questions which are not highlighted with EAT as well. Note that we test our systems on the Raw version of TrecQA test dataset.

#### 3.5.1 TrecQA

The current state of the art system is by [15] that uses EAT on word level model of [13]. Henceforth both results are presented. Our model RNN-Similarity on plain word level data fetches better result than the model of [13] by 2.24 % on MAP and 1.41 % on MRR. Our EAT words (single type), EAT words (different types) and EAT

<sup>6</sup><https://github.com/rsanjaykamath>



**Table 4.** Results reported on TrecQA, WikiQA, and SQUAD-Sent datasets. SQUAD-Sent dataset is a modified version for answer sentence selection task. RNN-S is RNN-Similarity model.

Datasets	Method	Acc.@1	MAP	MRR
TrecQA	Plain words - [13]	-	78	83.4
	EAT words - [15]	-	83.6	86.2
	Plain words - RNN-S	78.95	80.24	84.81
	EAT words (single type) - RNN-S	85.26	85.28	<b>89.16</b>
	EAT words (different types) - RNN-S	85.26	<b>85.48</b>	88.11
	EAT words (MAX+different types) - RNN-S	<b>86.32</b>	85.42	88.86
WikiQA	Plain words - [13]	-	70.9	72.3
	Plain words - [16]	-	<b>75.59</b>	<b>77.00</b>
	Plain words - RNN-S	56.79	69.07	70.55
	EAT words (single type) - RNN-S	56.38	68.63	70.59
	EAT words (different types) - RNN-S	58.4	<b>70.04</b>	<b>71.56</b>
	EAT words (MAX+different types) - RNN-S	57.20	69.17	70.89
SQUAD-Sent	Plain words - Implementation <sup>7</sup> of model by [13]	-	-	58.08
	Plain words - RNN-S	83.94	-	90.5
	EAT words (single type) - RNN-S	84.21	-	90.65
	EAT words (different types) - RNN-S	<b>84.26</b>	-	<b>90.70</b>
	EAT words (MAX+different types) - RNN-S	84.24	-	90.69

words (MAX + different types) models outperforms the state of the art performance for both MAP (1.68%) and MRR (2.96%) scores of the previous state of the art model by [15] where the MAP and MRR scores are higher for correct sentences being ranked as top 1.

### 3.5.2 WikiQA

Although a recent model by [16] which uses kernel methods outperforms all the scores of our model, we note that the performance on our EAT level models is higher than the ones on plain words. Only a few number of entities are annotated by spotlight compared to other datasets which is shown in the table 3. To annotate entities better we experimented using Spacy NER types directly which resulted in more annotated entities but reduced the performance lower than the word level scores.

### 3.5.3 SQUAD-Sent

SQUAD official test set is hidden to the public users. Although the difference between word

level and EAT word level is little, the difference highlights the fact that the entity words replaced in the sentence would not worsen the performance of the systems; instead it improves it subtly. We would like to note that the MAP and MRR values were the same because of the existence of just 1 positive sentence amongst other negative per question. Hence we only report MRR on this dataset. Plain words - [13] performance is obtained using the implementation available online<sup>8</sup>, which we experimented on SQUAD-Sent dataset.

One aspect to be highlighted is that the implementation<sup>8</sup> of word level model by [13] originally made for TrecQA dataset performs poorly (58.05%) SQUAD-Sent dataset (maybe because SQUAD-Sent has only one positive answer sentence per question whereas other datasets have several ones) which motivated us to build a model (RNN-Similarity) which works robustly for all the three datasets we have experimented with, without changing any specific hyperparameters of these models. Table 5 shows various results

<sup>8</sup><https://github.com/castorini/Castor>

**Table 5.** Results reported on TrecQA and SQUAD-Sent datasets using RNN-Similarity model trained only on EAT annotated questions

Datasets	Method	Acc.@1	MAP	MRR
TrecQA (EAT)	EAT words (single type)	84.15	84.81	87.17
	EAT words (different types)	85.37	85.45	88.18
	EAT words (MAX+different types)	<b>85.37</b>	<b>85.06</b>	<b>89.20</b>
WikiQA (EAT)	EAT words (single type)	<b>58.02</b>	<b>68.91</b>	<b>70.99</b>
	EAT words (different types)	55.14	67.70	69.52
	EAT words (MAX+different types)	56.38	68.16	69.83
SQUAD-Sent (EAT)	EAT words (single type)	83.81	-	90.53
	EAT words (different types)	84.04	-	90.61
	EAT words (MAX+different types)	<b>84.16</b>	-	<b>90.73</b>

on TrecQA and SQUAD-Sent datasets with only the questions which are annotated with EAT information in the train and test sets.

Training datasets with questions which contain EAT information only; if the question does not have a EAT value, it is discarded from the dataset below are the set of experiments and results:

- TrecQA (EAT): Apart from EAT words (MAX + different types) version of the dataset, the other two methods outperform word level models and EAT word level by [15] where the dataset statistics of this method can also be found.
- SQUAD-Sent(EAT): There is a difference of 8,800 questions from SQUAD-Sent dataset, which is considerably a huge number of missing questions. Yet the results from these experiments, do not decrease a lot, but rather perform better than SQUAD-Sent's plain word level model compare to EAT (different types) data.
- WikiQA (EAT): We remove the questions with 'NO-EAT' class which were 23 questions overall. The results are better with EAT (single type) which shows that the method works well in certain cases better than different types of EAT.

The results reported in table 5 shows that there is not a significant improvement over different methods when trained only on questions with

EAT information. Henceforth it is better to train models with the entire dataset and highlight EAT information only when the question contains the EAT information.

## 4 Conclusion and Future work

The Expected Answer Types are a useful piece of information that used to be extensively exploited in the traditional QA systems. Using them with the current state of the art DNN systems improves the system performance. We propose a simple model using recurrent neural networks which works robustly on three different datasets without any hyperparameter tuning and annotate entities belonging to the expected answer type of the question. Our model outperforms the previous state of the art systems in answer sentence task. We also propose a model to predict the expected answer type based on the question words using a multiclass classifier trained on a rule based system's output on a large scale QA dataset.

Future work involves using the expected answer types information in other downstream tasks such as in reading comprehension or multihop reading systems for extracting a short answer span.

## Acknowledgements

We would like to thank Harish Tayyar Madabushi from the University of Birmingham for providing us with the annotated questions for SQUAD dataset.

This work is funded by the ANR project GoAsQ (ANR-15-CE23-0022).

## References

1. **Chen, D., Fisch, A., Weston, J., & Bordes, A. (2017).** Reading wikipedia to answer open-domain questions. *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*.
2. **Daiber, J., Jakob, M., Hokamp, C., & Mendes, P. N. (2013).** Improving efficiency and accuracy in multilingual entity extraction. *Proceedings of the 9th International Conference on Semantic Systems*, ACM.
3. **Devlin, J., Chang, M.-W., Lee, K., & Toutanova, K. (2018).** Bert: Pre-training of deep bidirectional transformers for language understanding. *arXiv preprint arXiv:1810.04805*.
4. **He, H., Gimpel, K., & Lin, J. (2015).** Multi-perspective sentence similarity modeling with convolutional neural networks. *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing*.
5. **Kamath, S., Grau, B., & Ma, Y. (2018).** Verification of the expected answer type for biomedical question answering. *Companion Proceedings of the The Web Conference 2018*, WWW '18.
6. **Kim, Y. (2014).** Convolutional neural networks for sentence classification. *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*.
7. **Kolomiyets, O. & Moens, M.-F. (2011).** A survey on question answering technology from an information retrieval perspective. *Information Sciences*, Vol. 181, No. 24.
8. **Li, X. & Roth, D. (2006).** Learning question classifiers: the role of semantic information. *Natural Language Engineering*, Vol. 12, No. 3.
9. **Madabushi, H. T. & Lee, M. (2016).** High accuracy rule-based question classification using question syntax and semantics. *Proceedings of COLING 2016, the 26th International Conference on Computational Linguistics: Technical Papers*.
10. **Pennington, J., Socher, R., & Manning, C. (2014).** Glove: Global vectors for word representation. *Proceedings of the 2014 conference on empirical methods in natural language processing (EMNLP)*.
11. **Peters, M., Neumann, M., Iyyer, M., Gardner, M., Clark, C., Lee, K., & Zettlemoyer, L. (2018).** Deep contextualized word representations. *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers)*.
12. **Rajpurkar, P., Zhang, J., Lopyrev, K., & Liang, P. (2016).** Squad: 100,000+ questions for machine comprehension of text. *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing*.
13. **Rao, J., He, H., & Lin, J. (2016).** Noise-contrastive estimation for answer selection with deep neural networks. *Proceedings of the 25th ACM International on Conference on Information and Knowledge Management*, CIKM '16.
14. **Severyn, A. & Moschitti, A. (2015).** Learning to rank short text pairs with convolutional deep neural networks. *Proceedings of the 38th international ACM SIGIR conference on research and development in information retrieval*, ACM.
15. **Tayyar Madabushi, H., Lee, M., & Barnden, J. (2018).** Integrating question classification and deep learning for improved answer selection. *Proceedings of the 27th International Conference on Computational Linguistics*, Association for Computational Linguistics.
16. **Tymoshenko, K. & Moschitti, A. (2018).** Cross-pair text representations for answer sentence selection. *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing (EMNLP)*.
17. **Yang, Y., Yih, W.-t., & Meek, C. (2015).** Wikiqa: A challenge dataset for open-domain question answering. *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing*.
18. **Yin, W., Schütze, H., Xiang, B., & Zhou, B. (2016).** Abcnn: Attention-based convolutional neural network for modeling sentence pairs. *Transactions of the Association for Computational Linguistics*, Vol. 4.
19. **Yu, A. W., Dohan, D., Le, Q., Luong, T., Zhao, R., & Chen, K. (2018).** Fast and accurate reading comprehension by combining self-attention and convolution. *International Conference on Learning Representations*.

Article received on 29/01/2019; accepted on 04/03/2019.  
Corresponding author is Sanjay Kamath.



# A Fuzzy Approach to Mute Sensitive Information in Noisy Audio Conversations

Imran Sheikh<sup>1</sup>, Balamallikarjuna Garlapati,<sup>2</sup> Srinivas Chalamala,<sup>2</sup> Sunil Kumar Kopparapu<sup>1</sup>

<sup>1</sup> TCS Research and Innovation - Mumbai,  
Tata Consultancy Services,  
India

<sup>2</sup> TCS Research and Innovation - Hyderabad,  
Tata Consultancy Services,  
India

{imran.as, balamallikarjuna.g, chalamala.srao, sunilKumar.kopparapu}@tcs.com

**Abstract.** Audio conversation between a service seeking customer and an agent are common in a voice based call center (VbCC) and are often recorded either for audit purposes or to enable the VbCC to improve their efficiency. These audio recordings invariably contain personal information of the customer, often spoken by the customer to confirm their identity to get personalized services from the agent. This private to a person (P2aP) information is the recordings is a serious concern from the GDPR perspective and can lead to identity theft among other things. In this paper, we propose a robust framework that enables us to reliably spot the P2aP information in the audio and automatically mute it. The main contribution of this paper is the proposal of a novel fuzzy criteria which by design allows for reduced false alarms and at the same time increases the accuracy of the muting process even when the the speech to text conversion process is erroneous. Evaluation on real call center conversations demonstrates the reliability of the proposed approach.

**Keywords.** Fuzzy-muting, masking audio.

## 1 Introduction

Private to a Person (P2aP) Information can easily lead to identity theft if recorded audio conversations, between agents and customers in a call center, get into wrong hands. And more recently, the General Data Protection Regulation (GDPR) [5] makes it strict for an enterprise to

AGT: */how are you today/*

CUS: */i want to know my err my credit card dues/*

AGT: */can i have your name please/*

CUS: */~~ralf~~ ralf edison/*

AGT: */mr ~~edison~~. can i confirm your mobile number?/*

CUS: */sure it is ~~zero double one nine nine six two~~/*

AGT: */thank you. please confirm your credit card number/*

CUS: */my credit card number is ~~five three one six nine seven seven four one six nine seven eight two five zero~~/*

**Fig. 1.** Sample audio conversation with P2aP information to be muted struck off

refrain from retaining P2aP information. For these reasons, an enterprise, would like to *mute* all confidential (P2aP) information in their audio recordings, while retaining the rest of the conversation for audit or other purposes. Given an audio conversation (see Fig. 1), a muting approach should mute all instances of P2aP information while retaining the rest of the audio as is.

A typical P2aP information muting process uses a speech to text (S2T) or a key word spotting (KWS) engine to "locate" all the instances of P2aP information in the audio and then mute (erase) those portions in the audio. The process of locating P2aP information is dependent on the accuracy of the S2T or KWS engine. While the accuracy of S2T is getting better it is not yet perfectly reliable for conversational natural language speech as is the case during an Agent Customer interaction [9]. Noisy transcriptions, which are inherent in a S2T conversion process lead to false positives (Type I errors — non P2aP information are marked as P2aP), as well as false negatives (Type II errors — actual occurrences of P2aP are missed being identified).

Protection and masking of P2aP information has been widely studied in the fields of data security and data mining [12, 2]. Privacy of human speech in spaces like buildings, offices, hospitals, etc. has also been studied for a long time [4]. The focus, however has been to maintain a degree of privacy between the speaker and the intended listener, independent of the spoken content. Most methods available to mute P2aP information focus on (a) locating P2aP information in the audio conversation and then (b) muting the located regions [8, 11, 3, 7, 6, 17] in the audio. Almost all of them use some kind of S2T or KWS or acoustic pattern matching technique. Some of the more recent techniques focus on pattern matching in erroneous S2T hypothesis [7]. In all cases a binary decision is taken (mute or not-to-mute) based on a threshold associated with the located P2aP information inferred by the S2T engine. This binary decision works well for a S2T process which is accurate, however as mentioned earlier, the S2T process especially for natural spoken conversations is far from perfect.

In this paper, we introduce the concept of fuzzy-muting, a technique which is robust even when the S2T conversion process is not perfect. This is the main contribution of this paper. The rest of the paper is organized as follows Section 2 we dwell on the problems in existing P2aP information muting approaches. We present our approach in Section refsec:approach.

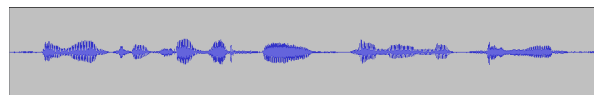


Fig. 2. Audio signal

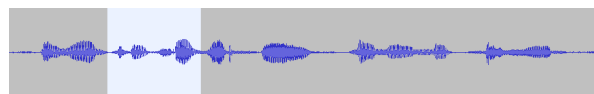


Fig. 3. Portion of the audio signal marked to be muted

Section 4 presents experimental analysis on real audio conversations and we conclude in Section 5.

## 2 Muting Problems

The basic process of muting P2aP information in audio involves three steps, namely, Step (a) – identify the set of patterns that are representative of the P2aP information, Step (b) – a mechanism to locate these patterns in the actual audio conversation, and then Step (c) – replace the located regions in the audio signal by silence or white noise (as depicted in Fig. 4).

Clearly Step (a) is a preparatory step and typically involves identifying a set of keywords or key-phrases and their possible occurrence patterns for a given conversation scenario and the Step (b) is the audio to text conversion process using a S2T (ASR) or KWS engine. The S2T engine gives a string of recognized words or labels and their corresponding confidence scores ( $s_c$ ). This confidence score plays a major role in muting P2aP information in existing approaches. There are several issues that crop up. We enumerate them below.

**Problem # 1** Speech to text (S2T) process is inherently erroneous for natural language conversational speech, subsequently these lead to false positives (Type I error) and false negatives (Type

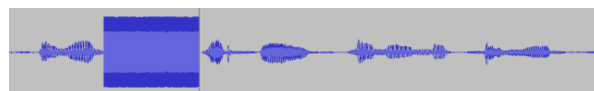


Fig. 4. Muting of the audio signal marked in Fig. 3

II error) in locating P2aP information instances. To overcome the Type I errors existing techniques set a threshold value ( $\tau$ ) on the log likelihood confidence score determined by the S2T engine and mute all the instances of words whose log likelihood score  $s_c \geq \tau$  and which corresponding to the P2aP information patterns as determined in Step (a). If  $t_s$  denotes the start time of the identified P2aP word or phrase and  $t_d$  denotes the duration of the P2aP word and if  $s_c$  is the log likelihood score of the P2aP word or phrase as determined by the S2T engine then muting of the signal  $x(t)$  is performed as follows:

$$\begin{aligned} &\text{if } (s_c > \tau) \quad x(t_s, t_s + t_d) = 0, \\ &\text{else } (s_c \leq \tau) \quad x(t_s, t_s + t_d) = x(t_s, t_s + t_d). \end{aligned} \quad (1)$$

Note that a high threshold ( $\tau$ ) could lead to false negatives (P2aP information not muted). So the choice of  $\tau$  becomes very crucial and its choice is dependent on the performance of the S2T engine.

**Problem # 2** The S2T conversion process can mis-recognize an audio segment corresponding to P2aP information into a text string or label which does not match the P2aP information patterns at all. For example, /one four two/ could be recognized by the S2T engine as "own for too". Irrespective of the confidence ( $s_c$ ) of the S2T, there is no way this kind of error can be handled.

**Problem # 3** A system based on threshold selection leads to a binary decision, namely, the muting process either renders the identified region muted or leaves it as it is (see (1)). This can lead to non-P2aP regions in an audio recognized by high confidence by the S2T engine to be muted as well as true P2aP regions recognized with lower confidence to be left untouched in the audio conversations.

We now propose an approach which is able to handle all the three problems associated with the conventional approaches to mute P2aP information in an audio conversation.

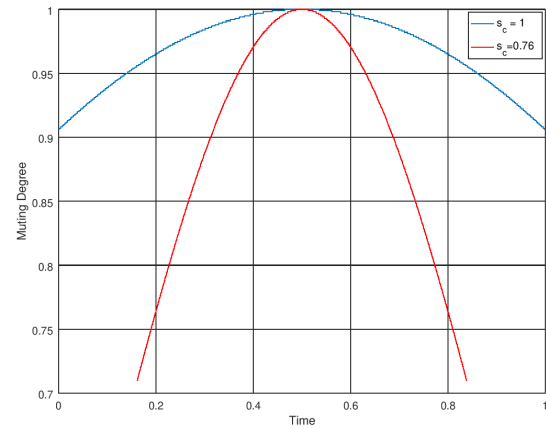


Fig. 5. (a)  $\mathcal{F}$  for different confidence scores ( $s_c$ ) (2)

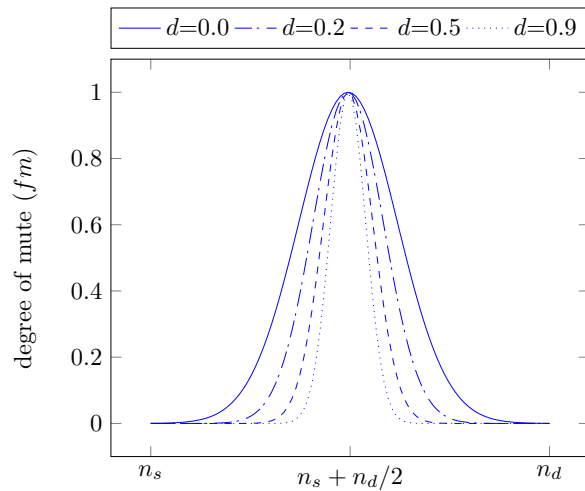
### 3 Proposed Approach

Our approach is mainly focused on addressing the problems in the existing systems enumerated earlier and as a result we are able to robustly mute P2aP information even in the presence of S2T conversion errors. To address Problem #1 and Problem #3 we replace the "binary" dependency on choice of the threshold ( $\tau$ ) with a non-binary *fuzzy-muting* using the log-likelihood scores ( $s_c$ ) determined by the S2T engine. To tackle Problem #2 we use a P2aP *search expansion* technique, which was originally proposed in [1].

#### 3.1 Fuzzy-Muting

Fuzzy-muting, is a non-binary muting operation which allows for decisions that are not hard. The effect of this is that this enables a *usable* robust muting solution even when the S2T conversion process is not perfect. The fuzzy-muting operation is a function of the confidence score  $s_c$  and does not explicitly depend on a threshold  $\tau$ . An example (used in our experiments) of a fuzzy-muting function (muting degree) is a Gaussian window, namely:

$$\mathcal{F}(t, t_d, s_c) = \exp \left\{ -\frac{\left( t - \frac{t_d}{2} \right)^2}{2s_c^2} \right\}, \quad (2)$$



**Fig. 6.** Fuzzy muting function  $\mathcal{F}^d$  for confidence score  $s_c = 0.5$  and different values of normalized edit distance  $d$  (4)

where,  $0 \leq t \leq t_d$  (for purpose of computation). A plot of this muting function is shown in Fig. 5. Note that when the confidence,  $s_c$ , of the recognized P2aP information is high then the muting function has a higher degree of mute and will mute most of  $(t_s, t_s + t_d)$  interval. Similarly, when the confidence,  $s_c$ , is low it takes the form of a peaky Gaussian, meaning only a small portion of the audio around  $t_s + t_d/2$  is muted and the rest remains the same. Thus  $\mathcal{F}$  does the following:

- P2aP information identified with high confidence (true positive) is muted heavily (blue curve, Fig. 5).
- P2aP information identified with low confidence (false positive) is muted for only a small portion of the identified duration, so that a non-P2aP information is not completely muted (red curve, Fig. 5).

### 3.2 Search Expansion

P2aP information regions in the audio signal can be missed due to mis-recognition by the S2T conversion process. In this case the S2T (or KWS) replaces the actual word with other word(s) from

the S2T vocabulary (or keyword list) which are phonetically similar. For example, a sequence of numbers */one two six four two/* could be mis-recognized as "one two six photo". Note that "four two" and "photo" are acoustically similar.

Such instances, in our technique are handled in the initial step of constructing P2aP information patterns. In our example, since */four two/* is expected as P2aP information, */photo/* which is acoustically similar is also included as a P2aP information cohort pattern.

When P2aP information cohorts are observed in the S2T output they can be mapped back and considered as confidential information. This mapping can lead to a correct identification of P2aP information or it leads to a new false positive. Either of these cases ultimately undergo fuzzy-muting as discussed earlier and not binary muting as in conventional muting systems.

To incorporate the proposed search expansion and the P2aP information cohort patterns the fuzzy muting function ( $\mathcal{F}$ ) can be updated to:

$$\mathcal{F} \rightarrow \mathcal{F}^d(t, t_d, s_c, d), \quad (3)$$

where,  $d$  is the normalized edit distance [13] between the phonemic pronunciation of the expected pattern (*/four two/* in our example) and the observed cohort pattern (*/photo/* in our example). This is incorporated into  $\mathcal{F}$  as:

$$\mathcal{F}^d(t, t_d, s_c, d) = \exp^{-\frac{\left(\left(t - \frac{t_d}{2}\right)(1+d^{0.5})\right)^2}{2s_c^2}}. \quad (4)$$

In this case, when the edit distance  $d$  is zero, i.e. the expected pattern and observed cohort pattern are same, the fuzzy muting function  $\mathcal{F}^d$  is same as that in  $\mathcal{F}$  (2) and directly dependent on the confidence score  $s_c$ . As the edit distance  $d$  increases, i.e. the expected pattern and observed cohort pattern become increasingly different,  $\mathcal{F}^d$  reduces the degree of mute. This is illustrated in Fig. 6.

Note that as  $d$  increases (less acoustic similarity between the actual word and the cohort) the muting becomes narrow, meaning only a small portion of the identified audio segment is muted.



## 4 Experimental Analysis

### 4.1 Dataset

For our experiments we use a dataset consisting of real telephone recorded conversations between customers and call center agents of an insurance company operating in the USA. The conversations are in US accented English and the duration of the conversations vary from 1 minute to 25 minutes. For our experiments, we considered the customer's social security number (SSN), as the P2aP information to be muted in these audio conversations. Note that other information, like name of the customer can also be included in the P2aP information to be muted. Of the 50 audio conversations in this dataset, 80% of them contain at least one instance of the P2aP information the rest 20% do not contain any instances of the P2aP information. With these characteristics, our dataset closely resembles a realistic P2aP information muting scenario.

### 4.2 S2T Setup

Further we use a S2T engine to locate all the instances of P2aP information in the described audio dataset. The public domain Kaldi ASR toolkit [16] is used with the ASplRE Chain acoustic models [14, 15] trained on conversational telephone speech from the Fisher English corpus. The accompanying pre-trained language model is used without any adaptation to the domain of our dataset. Given the un-adapted acoustic and language models, an average Word Error Rate (WER) of 47.2% is obtained on our dataset [10]. S2T confidence scores ( $s_c$ ) were generated from the lattice posterior probabilities using a language weight of 1.

### 4.3 S2T Conversion Error and Confidence Analysis

A total of 3757 P2aP tokens were actually spoken in the conversations. Of these, 65.6% were correctly recognized, 23.4% were substituted by another (cohort) word in the S2T lexicon and 10.8% were deleted (not recognized) by the S2T. One can conclude that there is about 34.2% chance that

the expected P2aP information will be in error and will affect a typical P2aP information muting process. The presence of 23.4% of substitution errors in P2aP tokens also indicate that the S2T engine missed these spoken P2aP words and replaced them by some other in-vocabulary word. Similarly, out of the 2860 P2aP tokens hypothesized by the S2T, 86.25% were correctly recognized tokens, 11.22% were substitutions to other words and 2.5% were insertions by the S2T. From this one can derive that about 13.72% of the hypothesized P2aP tokens are false positives and would affect the P2aP information muting process. These observations justify the need to address the problems described in Section 3 and also the need for search expansion.

For a more detailed analysis we present Fig. 7 which shows a distribution of S2T confidence scores for all the P2aP tokens hypothesized by the S2T as well as those originally present in the audio conversations but are substituted by the S2T. Instances where the S2T hypothesizes a P2aP token in place of another word are denoted by Substitutions (I) as they are Type I errors or false positives. Instances where the S2T misses a P2aP token and hypothesizes another word in place of it are denoted by Substitutions (II) as they are Type II errors or false negatives. The first observation from Fig. 7 is that although majority of the correctly hypothesized P2aP tokens have high confidence scores, setting a reasonable threshold like  $\tau = 0.8$  would discard at least 7.5% of the correctly hypothesized P2aP tokens which could be part of expected P2aP information. This re-confirms the need to address the Problems #1 and #3 described in Section 3 and the need for a fuzzy muting mechanism.

### 4.4 Muting Performance

We present an evaluation of the P2aP information muting approaches on our dataset of real call center calls, where SSNs are the P2aP information to be muted. In this evaluation our proposed approach comprises the fuzzy muting function (4). It is compared to a typical P2aP information muting approach which mutes S2T recognized P2aP tokens having a confidence score above a

preset threshold  $\tau$  (see (1)) to do a binary muting. Additionally we include a third approach in our evaluation, which first finds cohort patterns for P2aP sequences and then applies the typical P2aP information muting approach. In this approach, the confidence score on a P2aP token is modified as:

$$s'_c = \frac{s_c}{(1 + d^{\frac{1}{2}})^{\frac{1}{2}}},$$

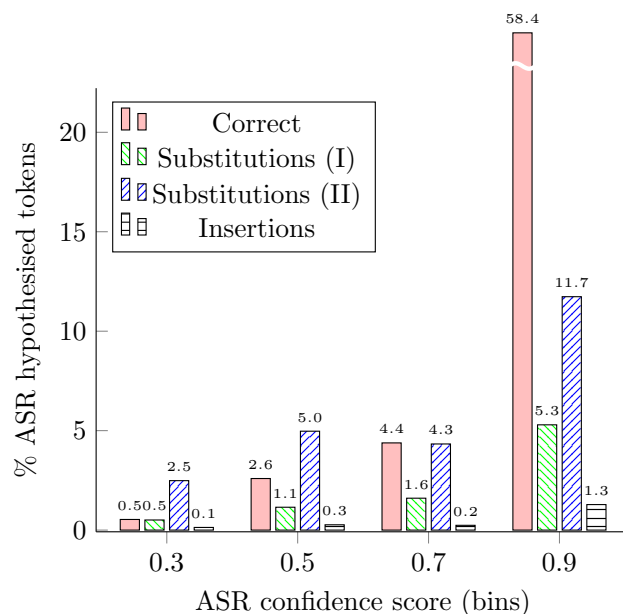
using the analogy between (2) and (4).

Our evaluation measure is the average number of SSN digit tokens which are left un-muted. A human listener listens to the muted speech signal and counts the number of SSN digit tokens left un-muted. The listener is allowed to listen to a muted signal multiple times and also to make guesses. The listener was given only those regions of the audio conversation where the user speaks the SSN sequence and not the complete audio conversation because of data confidentiality.

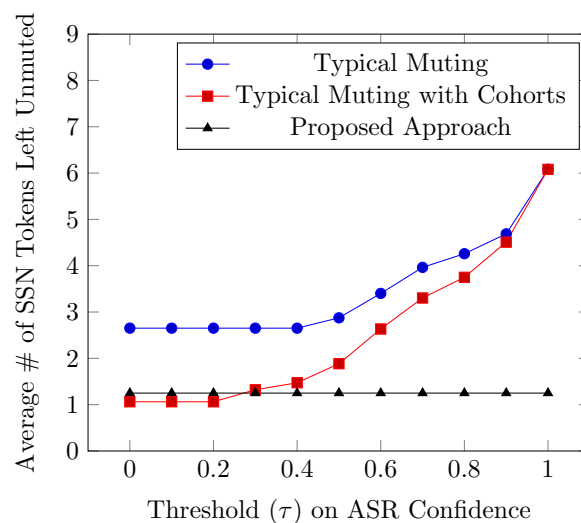
Figure 8 presents an evaluation and comparison of the P2aP information muting approaches. It shows a plot of the average number of P2aP tokens which are left un-muted by a muting approach, against different thresholds ( $\tau$ ) on S2T confidence scores. Approaches based on typical muting, (1), are dependent on the threshold ( $\tau$ ) and as the threshold ( $\tau$ ) is increased more number of P2aP tokens are left un-muted. Smaller thresholds ( $\tau$ ) appear better but they lead to more false alarms (instances of muting when there was no P2aP information) in other parts of the audio. Typical muting with cohorts gives a better performance on smaller thresholds but it degrades as the threshold is increased. Proposed approach comprising the fuzzy muting function (4) does not rely on the confidence threshold ( $\tau$ ) and achieves a significantly better muting performance, where on an average only 1.25 out of 9 SSN digits (P2aP) are left un-muted (see Fig. 8).

## 5 Discussion and Conclusion

Private to a person (P2aP) information often occur in recorded call center conversations during the process of customer verification which is required for the enterprise to provide personalized services to the customer.



**Fig. 7.** Distribution of S2T confidence scores for P2aP. (I indicates false positives from S2T and II indicates missed P2aP)



**Fig. 8.** Comparison of muting approaches. (Proposed Approach comprises fuzzy muting with cohorts)

Enterprises have the need to mute these information for several reasons including government regulations, for example, GDPR. Current muting solutions highly rely on the performance of a S2T engine. While S2T engines perform well in general, they fail to be 100% right when decoding natural spontaneous conversations [9]. In this paper, we have proposed a simple yet effective fuzzy muting function that can work even in the scenario when the S2T is not accurate all the time. The advantage of this approach is that it neither mutes non-P2aP information completely nor does it leave any potential P2aP information (including acoustically similar sounding) untouched, especially when the confidence in recognizing the P2aP information is low. Analysis on real call center conversations justifies the need for a fuzzy approach and the experiments demonstrate the effectiveness of the proposed approach. While in this work we resorted to Gaussian window functions, fuzzy muting could employ more sophisticated muting functions. This includes functions in which degree of mute is dependent on the envelope of the speech signal in the hypothesized region among other functions. Exploration and analysis in this direction is part of our future work.

## References

1. Ahmed, I. & Kopparapu, S. (2013). Improved method for keyword spotting in audio. *Proc. Acoustics*, New Delhi, India, pp. 1028–1033.
2. Aldeen, Y. A. A. S., Salleh, M., & Razzaque, M. A. (2015). A comprehensive review on privacy preserving data mining. *SpringerPlus*, Vol. 4, No. 1, pp. 694.
3. Bundock, D. S. & Ashton, M. (2008). System for excluding unwanted data from a voice recording. <https://www.google.com/patents/US20080221882>. US 20080221882 A1.
4. Cavanaugh, W. & Hirtle, P. (2006). Speech privacy in buildings: A review. *The Journal of the Acoustical Society of America*, Vol. 119, No. 5, pp. 3325–3325.
5. Commission, E. (2018). 2018 reform of EU data protection rules. [https://ec.europa.eu/commission/priorities/justice-and-fundamental-rights/data-protection/2018-reform-eu-data-protection-rules\\_en](https://ec.europa.eu/commission/priorities/justice-and-fundamental-rights/data-protection/2018-reform-eu-data-protection-rules_en).
6. Doren, G. K. (2013). Selective security masking within recorded speech. <https://www.google.co.in/patents/US8433915>. US 8433915 B2.
7. Faruque, T. A., Negi, S., & Subramaniam, L. V. (2009). Protecting sensitive customer information in call center recordings. *IEEE International Conference on Services Computing*, pp. 81–88.
8. Hillis, W. D., Ferren, B., & Howe, R. (2007). Method and system for masking speech. <https://www.google.com/patents/US7184952>. US 7184952 B2.
9. Kopparapu, S. K. (2014). *Non-Linguistic Analysis of Call Center Conversations*. Springer Publishing Company.
10. Kopparapu, S. K. (2014). Word error rate. <https://sites.google.com/site/nlccanalytics/home/wer>.
11. Lee, H., Lutz, S., & Odinak, G. (2007). Selective security masking within recorded speech utilizing speech recognition techniques. <https://www.google.co.in/patents/US20070016419>. US 20070016419 A1.
12. Lodha, S., Patwardhan, N., Roy, A., Sundaram, S., & Thomas, D. (2012). Data privacy using MASKETEER™. *Proceedings of the 9th International Conference on Theoretical Aspects of Computing, ICTAC'12*, Springer-Verlag, Berlin, Heidelberg, pp. 151–158.
13. Marzal, A. & Vidal, E. (1993). Computation of normalized edit distance and applications. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, Vol. 15, No. 9, pp. 926–932.
14. Peddinti, V., Chen, G., Manohar, V., Ko, T., Povey, D., & Khudanpur, S. (2015). JHU ASPIRE system: Robust LVCSR with TDNNs, iVector adaptation and RNN-LMS. *IEEE Workshop on Automatic Speech Recognition and Understanding (ASRU)*, pp. 539–546.
15. Povey, D. (2017). Kaldi models. <http://kaldi-asr.org/models.html>.
16. Povey, D., Ghoshal, A., Boulianne, G., Burget, L., Glembek, O., Goel, N., Hannemann, M., Motlicek, P., Qian, Y., Schwarz, P., Silovsky, J., Stemmer, G., & Vesely, K. (2011). The Kaldi speech recognition toolkit. *IEEE 2011 Workshop on Automatic Speech Recognition and Understanding*, IEEE Signal Processing Society, pp. 1–4. IEEE Catalog No.: CFP11SRW-USB.

- 17. Schachter, J. & Levin, K. D. (2013).**  
System and method for removing  
sensitive data from a recording.  
<https://www.google.com/patents/US20130266127>.  
US 20130266127 A1.

*Article received on 10/01/2019; accepted on 04/03/2019.*  
*Corresponding author is Imran Sheikh.*

# A Comparative Study on Text Representation Models for Topic Detection in Arabic

Rim Koulali<sup>1</sup>, Abdelouafi Meziane<sup>2</sup>

<sup>1</sup> Hassan II University, Faculty of Sciences Ain Chock, LIMSAD Laboratory, Morocco

<sup>2</sup> Mohammed I University, Sciences Faculty, LARI Laboratory, Morocco

rim.koulali@gmail.com, abdelouafi\_meziane@yahoo.com

**Abstract.** Topic Detection (TD) plays a major role in Natural Language Processing (NLP). Its applications range from Question Answering to Speech Recognition. In order to correctly detect document's topic, we shall first proceed with a text representation phase to transform the electronic documents contents into an efficiently software handled form. Significant efforts have been deployed to construct effective text representation models, mainly for English documents. In this paper, we realize a comparative study to investigate the impact of using stems, multi-word terms and named entities as text representation models on Topic Detection for Arabic unvowelized documents. Our experiments indicate that using named entities as text representation model is the most effective approach for Arabic Topic Detection. The performances of the two other approaches are heavily dependent on the considered topic. In order to enhance the Topic Detection results, we use combined vocabulary vectors based on stems and named entities (respectively stems and multi-word terms) association to model topics more accurately. This approach effectiveness has been endorsed by the enhancement of the system performances.

**Keywords.** Natural language processing, topic detection, text representation, multi-word terms, named entities.

## 1 Introduction

With the exponential growth of available Arabic electronic documents, important research effort is deployed to effectively manage, explore, retrieve and analyze the information they embody.

Topic Detection (TD) represents an important Natural Language Processing (NLP) and Information Retrieval (IR) task that has been employed to satisfy the users' information needs.

Topic Detection is a widely studied topic for western languages due to its application in many Information Retrieval (IR) and NLP tasks such as: social media content analysis [21], newspaper documents classification [28], Speech Recognition [29], Summarization [17], etc. Nevertheless, for Modern Standard Arabic (MSA), the research efforts are still limited as relatively few works have been carried. The Arabic language presents researchers and developers of Natural Language Processing (NLP) applications for Arabic text and speech with serious challenges. This is due to the complex morphology of the Arabic language and other characteristics such as the absence of diacritics, the lack of capitalisation and specially its highly inflectional nature.

Performing Topic Detection accurately depends essentially on the quality documents representation. A challenging characteristic of the Topic Detection problem is the extremely high dimensionality of text data that implies an effective text representation model. Therefore, text representation happens to be a crucial aspect of Topic Detection (TD) process. The process must allow coping with texts complexity and easing their manipulation by mapping them from the full textual version into a compact form of its content, in order

to give an effective document representation model to build an efficient Topic Detection system.

The problem we are treating in this work is to answer the following questions: What is a good representation of news documents? What is the best text representation? Is a good text representation enough to give better performances? Is a text representation model that reduces the feature space to a limited set of dimension effective? Can we develop a text representation that tackles with the various documents belonging to different topics?. The outcomes of our study are expected to compare the performance of the proposed text representation models, taking into account the aforementioned challenges (questions?). Therefore, we identify the text representation model that optimize the process of capturing textual patterns and maximize the Topic Detection system efficiency.

This work is mainly concerned with studying text representation models. Our objective here is to conduct a comparative study of three different text representation models, namely: stems, Multi-Word Terms (MWTs) and Named Entities (NEs) in order to evaluate their influence on the quality of Topic Detection systems. The use of NEs and MWTs for Arabic Topic Detection is motivated by the fact that the amount of information contained in a coalition of words is much important than the one of individual terms.

To the best of our knowledge, a comparative study between the three models has never been conducted before for Arabic Topic Detection. We have realized several experiments in which we firstly benchmarked the three models. Then, we have studied the use of weights vectors formed respectively by the combination of stems and named entities along with vectors formed by stems and multi-word Terms.

The reminder of this paper is structured as follows: Section II defines the main concepts studied in this paper which are Topic Detection and text representation. Section III, presents stems, multi-word terms and named entities as the used approaches for text representation for the Arabic Topic Detection. Experiments set up, evaluation metrics and results of benchmarked techniques are

detailed in Section IV. Section V concludes the paper and announces our future works.

## **2 Topic Detection and Text Representation**

### **2.1 Topic Detection**

The term topic is usually defined as the aboutness of a unit of discourse [22]. Topic Detection (TD) is a part of the study content of Topic Detection and Tracking (TDT). It is a new skill by which a given set of documents from the data stream, such as news reports, newswires, blogs and social media, are classified into a given set of documents into thematic categories.

In the following, we exclusively consider the single-Topic Detection version, since it is more general than multi-Topic Detection: the latter can be split into several binary (i.e., single-label) detection problems, but the contrary is not possible. Also, we are interested in using a fixed set of topics instead of an open one. Our technique allows the specification of topics of interest and attempts to classify documents within those topics only, based on two major steps: topic vocabulary generation and topic assignment.

TD is based on supervised or unsupervised learning using a training corpus to represent each topic with a specific model obtained using a wide range of text processing approaches, text representation models, and detection methods to estimate similarities between topics and documents vectors.

### **2.2 Text Representation**

Text representation is used to reduce the complexity of the documents, capture the meaning of them and make them easier to handle. Extensive work is carried out to propose various text representations. As definition of a document is that it is made of a joint membership of terms which have various patterns of occurrence. Thus it has to be converted from the full text version to a standard representation. Bag of Word (BoW) happens to be the basic representation model where a document is typically represented as a vector of term weights

(word features) from a set of terms, using the frequency count of each term in the document. This model of document representation is also called a Vector Space Model (VSM) [26].

However, the BOW/VSM representation has its own limitations, namely: the extremely high dimensionality of text data, loss of correlation with adjacent words and loss of semantic relationship that exist among the terms in a document [8]. To overcome these problems, we present and experiment in this paper many text representation models while trying especially to preserve semantic relations between words by using Named Entities or Multi-words Terms.

### 3 Adopted Text Representation Models

#### 3.1 Stems

Terms have many morphological variants that will not be recognized by term matching algorithm without additional text processing. In most cases, these variants have similar semantic interpretation and can be treated as equivalence in text mining. Stemming has become an important step in text mining and information retrieval in order to make the information more accurate and effective.

Stemming in Arabic language can be defined as the process of removing prefixes or/and suffixes from words to recover their stems. The aim of the Stemming or Light Stemming is not to produce the root of a given Arabic word, rather is to remove the most frequent suffixes and prefixes. However, till now there is almost no standard algorithm for Arabic light stemming, as there is no definite list of suffixes and prefixes that must be removed.

Experiments have shown that using stems is more efficient than roots or the full words in Arabic Topic Detection [16].

We used the morphological analyzer Alkhalil [9] to recover a list of stems for each document. Alkhalil realizes a morphological analysis for each word in the corpus and returns among other morphological information all possibly related stems to the considered word. So, we implemented a Viterbi algorithm to keep only the stems that are relevant to the context.

#### 3.2 Arabic Multi-Word Terms

Although multi-word term has no totally agreed upon definition, it can be understood as a sequence of two or more consecutive individual words, forming a semantic unit [24]. In fact, the exact meaning of a multi-word term could not be fully achieved by its individual parts.

MWTs Extraction represents an important task of Automatic term recognition and is employed in numerous NLP fields such as: Text Mining [27], Syntactic Parsing [20, 3], Machine Translation [11] and Text Classification [34]. The MWTs extraction task covers detection and extraction of a set of consecutive semantically related words and the techniques used can be classified into four major categories: (a) Statistical approaches based on frequency, probability and co-occurrence measures [31], (b) Symbolic approaches using parsers, morphological analysis, MWTs boundaries detection and patterns [33], (c) Hybrid approaches combining statistical and morphological methods [12] and (d) Word alignment approaches [18].

We built our multi-word extraction system based on the hybrid approach which performs in two steps:

- Linguistic filtering: The objective of the developed linguistic filter is to extract Multi-Word candidate terms. We preprocess the documents using the text processing method explained in section 4.2 without the stemming part. We use a Part-Of-Speech Tagger to assign morphological tickets to the corpus document's words using The Stanford Arabic POS Tagger [30]. This step will help us to detect possible MWTs following the patterns below:

Noun +.

- Noun; [Adjective]+.
- Noun; Preposition; Noun.

In order to extract multi-word terms, the document sentences are scanned for sets of words conform to one of the above listed patterns and ordered by their number of occurrences.

We consider only the sets of words appearing at least twice in each document. The linguistic filter allows extracting MWTs candidates with various sizes; Bigrams, Trigrams and Four grams.

- Statistical filtering: To reduce linguistic ambiguities and increase the ratio of correct extracted MWTs, we used two well-known methods for their high effectiveness in MWTs extraction, namely: C-value [14] for the nested words and their variations along with Log Likelihood Ratio(LLR) [13] to gather the remaining MWTs Bigrams.

The implemented MWTs extraction system achieved an overall of 90.25% in term of precision.

### 3.3 Arabic Named Entities

The objective of Named Entity Recognition (NER) task is to identify and classify mentions of rigid designators from text belonging to named entity types such as: persons, organizations, locations and miscellaneous names (date, time, percentage and monetary expressions) within an open-domain text [19]. The NER is a key technique of Information Extraction and Question Answering systems. The techniques proposed in the literature of NER fall within three major categories: (a) Rule-based approaches [23]: is a language dependent approach that uses hand crafted linguistic rules, (b) Machine learning based approaches [15]: is a language independent approach based on machine learning algorithms and (c) Hybrid approaches [10]: combines linguistic patterns and machine learning techniques.

We developed an Arabic NER system implementing the hybrid approach. We used ANERCorpus [7] for the training and test corpus which is an annotated corpus following the Conferences on the Computational Natural Language Learning (CoNLL) 2002 and 2003 shared task, formed by tags falling into the following four categories: Person, Location, Organization and Miscellaneous names. Also, we used a SVM based software for sequence tagging using Hidden Markov Models [2], along with a combination of language independent

and language specific binary features to capture the essence of the Arabic language such as: lexical, contextual, morphological features and gazetteers,... We boosted the system with an automatic pattern extraction framework in order to enhance the ANER system. The developed system achieved an overall of F1-measure of 83.20%.

## 4 Experiments and Evaluation

### 4.1 The Dataset

For the setup of our experiments, we used a corpus of over 20.291 articles, collected from the Arabic newspaper Wattan of the year 2004 [1]. The corpus contains articles covering the six following topics: culture, economics, international, local, religion and sport. The repartition of documents is described in Table 1. The corpus was divided into two subsets of documents. Thus, 9/10 of the corpus was dedicated to training the feature selection system (Topic vocabulary construction), whereas 1/10 of the overall documents formed the evaluation corpus.

**Table 1.** Number of documents per topic

Topics	Number of articles
Culture	2782
Economy	3468
International	2035
Local	3596
Religion	3860
Sports	4550

### 4.2 Experiments Setup

Text preprocessing is the first step in a Topic Detection process. It aims to reduce the noise in documents by removing all the unnecessary terms and mistyped words. We process the corpus using the following operations:



- Document pretreatment: this step covers the unification of documents encoding to avoid any ambiguity, along with the elimination of Latin words, symbols, numbers, Roman numeral and special characters.
- Word normalisation : words having the same meaning are normalized and represented in the same standard based on some specific rules related to the Arabic language to eliminate ambiguity and reduce the redundancy.
- Stop words elimination: stop words are considered to be information free words, thus their removal will not affect the Topic Detection system performances. We eliminate stop words by comparing each word with the elements of a hand crafted list containing over 600 stop words including: prepositions, demonstrative pronouns, identifiers, logical connectors,...

We use the preprocessed training corpus to generate Topic Representation Model for each of the three models presented in section 3 as follow:

- Stem representation: For each topic, we process all training corpus documents to extract stems for all the words. Then, we calculate the Mutual Information (IM) [6] value for each stem. Given a word  $w$  and a topic  $t$ , with respective individual occurrence probabilities and , the Mutual Information (MI) is expressed by the following equation :

$$IM(w, t) = \log(P(w|t)) - \log(P(w)). \quad (1)$$

After sorting, we retain words with higher MI values to build the vocabulary vector of features representing each topic. The vocabulary vectors are constructed by the TF-IDF [25] weights of the features within a predefined Mutual Information threshold. We adapted the classic TF-IDF to represent topics vectors. For instance, each Topic is represented by a vector that contains the weights of the topic vocabulary words. The weight  $t_{jk}$  of the  $k^{th}$  vocabulary word of topic  $j$  is expressed as follows:

$$\begin{cases} t_{jk} = n f_j^k * idf_k. \\ idf_k = \log(\frac{N}{df_k}). \end{cases} \quad (2)$$

where :  $n f_j^k$  denotes the frequency of the word  $k$  in documents of the training corpus belonging to topic  $j$ ,  $df_k$  is the number of topics in which the word  $k$  appears at least once,  $N$  is the total number of topics and  $idf_k$  represents the inverse document frequency.

- Multi-word terms representation: For each topic, the extracted MWTs are ranked by their LLR value if they are bigrams or C-value score if they are nested. We sorted the terms and built the vector vocabulary with the MWTs having higher scores using TF-ITF [5] score as a term weight. Given a set  $T$  of multi-word terms extracted from the topic  $i$ , the TF-ITF value of multi-word  $t \in T$  is:

$$\begin{cases} w_i(t) = \log(f_i(t)) - ITF(t). \\ ITF(t) = \log\left(\frac{\sum_{t \in T} F(t)}{F(t)}\right). \end{cases} \quad (3)$$

where  $f_i(t)$  and  $F(t)$  are respectively the frequencies of term  $t$  in the topic  $i$  and in the corpus.

- Named entities representation: We generated four categories of NE for each topic. We calculated the mutual information value of each NE and sorted them to extract vocabulary vectors formed by NE with the highest scores. Then, we used TF-IDF to weight each NE of each category. Each topic gets to be fully represented by one vector containing the four categories sorted by their TF-IDF score.

To evaluate the performance of the developed system, we process the test corpus to extract vectors representing each test document according to the preprocessing steps detailed earlier for each one of the three text representation models. Then, we compute similarity between each test document and the generated topics vocabulary vectors using Cosine similarity. The cosine similarity value lies within the real interval of [0; 1], where 1 indicates a perfect match between the two vectors and 0 indicates a complete mismatch. The cosine similarity formula is expressed as follows:

$$\cos(\theta) = \cos(T_j, D_i) = \frac{\sum_{k=1}^n v_{jk} * w_{ik}}{\sqrt{\sum_{k=1}^n v_{jk}^2} \times \sqrt{\sum_{k=1}^n w_{ik}^2}}, \quad (4)$$

where  $T_j = v_{j1}, v_{j2}, \dots, v_{jn}$  represents the  $j^{th}$  topic feature vector and  $D_i = w_{i1}, w_{i2}, \dots, w_{in}$  represents features vector of test document  $i$ . The test document is assigned to the topic with the maximum cosine similarity measure.

### 4.3 Evaluation Methods

In order to evaluate the classifiers performance, three standard set-based metrics are used: recall and precision [4]. For a given topic  $T_i$ , precision and recall are defined as follows:

$$Precision = \frac{\#documents \text{ in } T_i \text{ classified to } T_i}{\#documents \text{ classified to } T_i}, \quad (5)$$

$$Recall = \frac{\#documents \text{ in } T_i \text{ classified to } T_i}{\#documents \text{ in } T_i}. \quad (6)$$

The value of precision and recall often depend on parameter tuning; there is a tradeoff between them. This why we also use another measure that combine both of precision and recall: the F1-measure [32], defined as :

$$F1 - measure = \frac{2 \times Precision \times Recall}{Precision + Recall}. \quad (7)$$

We also calculate the macro average of each metric which is given by the average on the metric scores of all topics.

### 4.4 Results

Figure 1 shows the Topic Detection system performance results using Stems, MWTs and NEs on each topic of the test corpus in term of F1-measure. We notice that the named entities approach realizes higher performances. Among the six topics, all the three models gave the best performance on the "Sport" topic and the poorest performance on the "Culture" topic. This could be explained by the difficulty faced to extract truly representative terms from the training corpus since the "Culture" topic covers a broad range of documents including TV programs, movies, poetry, literature, culinary, museum events..., which is not the case of the "Sport" topic.

In general, the performance of MWTs and NEs remain globally more important than Stems. This is due to the influence of the Arabic language

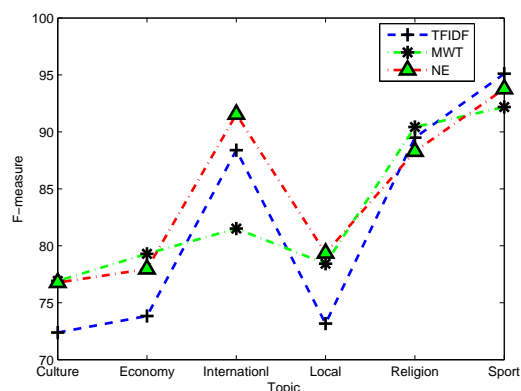


Fig. 1. F1-measure per topic

nature. In fact, one word can be used in many sentences with different meaning. For example, the word elected in Arabic can be used in local documents as an adjective referring to an elected person, and can also be used in sports documents as a noun referring a sport team. This ambiguity can be reduced by using MWTs or NEs as adding one or more words with give more precision about the context of the word. Although Stems outperforms MWTs on "International" topic, this can be explained by the fact that this topic's documents are generally formed by names of cities and countries composed with one word only.

Table 2 presents the global performance of the three approaches based on the macro-average of all measures across the six topics. It can be seen that NEs outperformed the other two approaches in precision, recall and F1-measure. MWTs achieved better performance than Stems in terms of precision and F1-measure.

Table 2. Topic Detection macro-average performance results

Models	Precision	Recall	F1-measure
Stems	82.29	82.40	82.35
MWTs	84.13	82.81	83.46
NEs	85.63	84.74	85.18

On one hand, this is due to the fact that the NEs and MWTs carry an important amount of information, which is very benefic to the Arabic Topic Detection rather than using stems only. On the other hand, we notice that the effectiveness of multi-word is strongly dependent on the types of literature. For instance, multi-words as a text representation is effective for documents, in which fixed expressions (terminologies, collocations, etc.) are usually used, such as academic papers, but may be not effective for the documents with extensive topics, in which fixed expressions are not usually used such as poems (culture).

This ambiguity generated in such cases affects the performances of the Topic Detection system, leading to a misjudgment of the similar topics. In fact, great many words are common to different topics leading to imprecision in the Topic Detection process. Although, it can help to differentiate similar topics by named entities, the number of named entities is limited in news documents. Using only named Entities; it may influence the topic detection system as many key words which describe the topics are ignored. As a result, the performance of Topic Detection decrease for related topic such as: culture and local as these topics show a low performance for the three models.

In order to solve the problems related to the difficulty in distinguishing similar topics, we investigate the use of topic oriented combined vocabulary vectors: Stems with NEs and Stems with MWTs. We calculate the similarity as follows:

$$Sim(d, t) = \begin{cases} \alpha sim(V_{mwt}^d, V_{mwt}^t) + \beta sim(V_{st}^d, V_{st}^t), \\ \alpha sim(V_{ne}^d, V_{ne}^t) + \beta sim(V_{st}^d, V_{st}^t), \end{cases} \quad (8)$$

where:

- $\alpha$  and  $\beta$  are weight factors with the constraint  $\alpha + \beta = 1$ .
- $V_y^x \in \{d, t\} \times \{MWTs, NEs, STs\}$  vectors containing MWTs, NEs and Stems weights for the considered document  $d$  and topic  $t$  respectively.

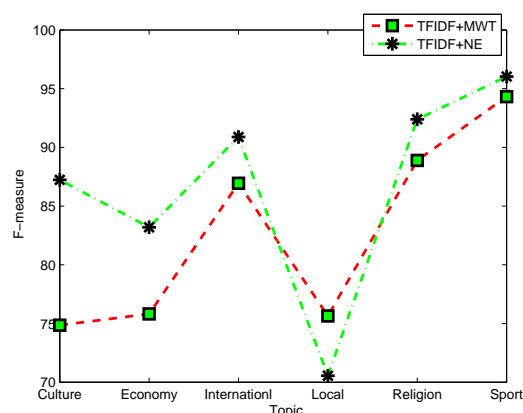


Fig. 2. F1-measure per topic for combined vectors

—  $Sim(V_y^t, V_y^d, y \in \{MWTs, NEs, STs\})$  stands for the Cosine Similarity between topic  $t$  and document  $d$  vectors for each text representation model.

The  $\alpha$  and  $\beta$  values were obtained empirically through running the experiments for various weight values and selecting those ensuring optimal performances, corresponding to  $\alpha = \frac{3}{4}$  and  $\beta = \frac{1}{4}$ . As shown in Figure 2, we can clearly notice that using the combined vectors approach has improved the system performances. This improvement is justified by the fact that the combined vectors allow expressing each topic more accurately and highlight the differences between similar topics. Nerveless, the results of combining stems with multi-words terms remain less efficient than those of associating stems with names entities.

Table 3. Topic Detection macro-average performance for combined vectors approach

Models	Precision	Recall	F1-measure
Stems+MWTs	84.40	84.10	84.25
Stems+NEs	89.27	88.33	88.80

Table 3 shows that combining stems and multi-word terms vectors has augmented the F1-measure from 83.46% to 84.25%.

The augmentation in the case of named entities categories and stems combined vectors is much important. Indeed, we have an average of F1-measure of 88.80% against 85.18% realized with named entities vectors only. We conjecture that the combination method is more effective and deliver better results.

## 5 Conclusions and Future Works

In this paper, we conducted several experiments to evaluate the performances of three text representation models namely: Stems, Multi-Word Terms and Named Entities in the context of Arabic Topic Detection. We conjecture that text representation based on Named Entities is very effective for TD with a performance variance related to the topic nature. This can be explained by the important amount of information contained in NEs.

The conducted experiments show also that MWTs have better performances in Topic Detection than Stems. We notice a variance in the MWTs performances depending on the topic's nature, some topics are described with an important amount of words composed with one gram which can explain the high performance of Stems in some topics over the MWTs.

To overcome the problem of similar topics distinguish, especially for the literature topics (culture, local), we ran experiments using combined vectors of Stems and named entities (respectively Stems and multi-word terms). The results were very significant and outperformed the earlier results obtained by using each one of the three models separately. The combination approach proved to be very effective by enhancing the overall system performance and taking into account the semantic relationship between words.

To improve the text representation models for a better Topic Detection process, we intend to use external dictionaries such as WordNet and ontologies to enhance the generation of the vocabulary vectors.

## References

1. **Abbas, M., Smaili, K., & Berkani, D. (2010).** Tr-classifier and knn evaluation for topic identification tasks. *the International Journal on Information and Communication Technologies (IJICT)*, Vol. 3, No. 3, pp. 65–74.
2. **Altun, Y., Tsochantaridis, I., Hofmann, T., et al. (2003).** Hidden markov support vector machines. *ICML*, volume 3, pp. 3–10.
3. **Attia, M. A. (2006).** Accommodating multiword expressions in an arabic lfg grammar. In *Advances in Natural Language Processing*. Springer, pp. 87–98.
4. **Baeza-Yates, R., Ribeiro-Neto, B., et al. (1999).** *Modern information retrieval*, volume 463. ACM press New York.
5. **Basili, R., Moschitti, A., Pazienza, M. T., & Zanzotto, F. M. (2001).** A contrastive approach to term extraction. *Terminologie et intelligence artificielle. Rencontres*, pp. 119–128.
6. **Battiti, R. (1994).** Using mutual information for selecting features in supervised neural net learning. *Neural Networks, IEEE Transactions on*, Vol. 5, No. 4, pp. 537–550.
7. **Benajiba, Y., Rosso, P., & Beneditruiz, J. M. (2007).** Anersys: An arabic named entity recognition system based on maximum entropy. In *Computational Linguistics and Intelligent Text Processing*. Springer, pp. 143–153.
8. **Bernotas, M., Karkliu, K., Laurutis, R., & Slotkienė, A. (2015).** The peculiarities of the text document representation, using ontology and tagging-based clustering technique. *Information Technology and Control*, Vol. 36, No. 2.
9. **Boudlal, A., Lakhouaja, A., Mazroui, A., Meziane, A., Ould Abdallahi Ould Bebah, M., & Shoul, M. (2011).** Alkhalil morphosys: Morphosyntactic analysis system for non vocalized arabic. *Seventh International Computing Conference in Arabic*.
10. **Chiticariu, L., Krishnamurthy, R., Li, Y., Reiss, F., & Vaithyanathan, S. (2010).** Domain adaptation of rule-based annotators for named-entity recognition tasks. *Proceedings of the 2010 conference on empirical methods in natural language processing*, Association for Computational Linguistics, pp. 1002–1012.
11. **Deksne, D., Skadins, R., & Skadina, I. (2008).** Dictionary of multiword expressions for translation into highly inflected languages. *LREC*.

12. Duan, J., Zhang, M., Tong, L., & Guo, F. (2009). A hybrid approach to improve bilingual multiword expression extraction. *Pacific-Asia Conference on Knowledge Discovery and Data Mining*, Springer, pp. 541–547.
13. Dunning, T. (1993). Accurate methods for the statistics of surprise and coincidence. *Computational linguistics*, Vol. 19, No. 1, pp. 61–74.
14. Frantzi, K. T. & Ananiadou, S. (1996). Extracting nested collocations. *Proceedings of the 16th conference on Computational linguistics-Volume 1*, Association for Computational Linguistics, pp. 41–46.
15. Ju, Z., Wang, J., & Zhu, F. (2011). Named entity recognition from biomedical text using svm. *2011 5th international conference on bioinformatics and biomedical engineering*, IEEE, pp. 1–4.
16. Koulali, R. & Meziane, A. (2011). Topic detection in arabic unvowelized documents. *Proceedings of the ALTIC*, pp. 54–58.
17. Lloret, E. (2009). Topic detection and segmentation in automatic text summarization.
18. Moirón, B. V. & Tiedemann, J. (2006). Identifying idiomatic expressions using automatic word-alignment. *Proceedings of the EACL 2006 Workshop on Multi-wordexpressions in a multilingual context*, pp. 33–40.
19. Nadeau, D. & Sekine, S. (2007). A survey of named entity recognition and classification. *Lingvisticae Investigationes*, Vol. 30, No. 1, pp. 3–26.
20. Nivre, J. & Nilsson, J. (2004). Multiword units in syntactic parsing. *Proceedings of Methodologies and Evaluation of Multiword Units in Real-World Applications (MEMURA)*.
21. Quercia, D., Askham, H., & Crowcroft, J. (2012). Tweetlda: supervised topic classification and link prediction in twitter. *Proceedings of the 3rd Annual ACM Web Science Conference*, ACM, pp. 247–250.
22. Renkema, J. (2004). *Introduction to discourse studies*. John Benjamins Publishing.
23. Riaz, K. (2010). Rule-based named entity recognition in urdu. *Proceedings of the 2010 named entities workshop*, Association for Computational Linguistics, pp. 126–135.
24. Sag, I. A., Baldwin, T., Bond, F., Copestake, A., & Flickinger, D. (2002). Multiword expressions: A pain in the neck for nlp. *International Conference on Intelligent Text Processing and Computational Linguistics*, Springer, pp. 1–15.
25. Salton, G. (1991). Developments in automatic text retrieval. *science*, Vol. 253, No. 5023, pp. 974–980.
26. Salton, G., Wong, A., & Yang, C.-S. (1975). A vector space model for automatic indexing. *Communications of the ACM*, Vol. 18, No. 11, pp. 613–620.
27. SanJuan, E. & Ibekwe-SanJuan, F. (2006). Text mining without document context. *Information Processing & Management*, Vol. 42, No. 6, pp. 1532–1552.
28. Schwartz, R., Imai, T., Kubala, F., Nguyen, L., & Makhoul, J. (1997). A maximum likelihood model for topic classification of broadcast news. *Fifth European Conference on Speech Communication and Technology*.
29. Torres, R., Kawanami, H., Matsui, T., Saruwatari, H., & Shikano, K. (2012). Topic classification of spoken inquiries using transductive support vector machine.
30. Toutanova, K., Klein, D., Manning, C. D., & Singer, Y. (2003). Feature-rich part-of-speech tagging with a cyclic dependency network. *Proceedings of the 2003 Conference of the North American Chapter of the Association for Computational Linguistics on Human Language Technology-Volume 1*, Association for Computational Linguistics, pp. 173–180.
31. Van de Cruys, T. & Moirón, B. V. (2007). Lexico-semantic multiword expression extraction. *Proceedings of the 17th Meeting of Computational Linguistics in the Netherlands (CLIN)*, pp. 175–190.
32. Van Rijsbergen, C. J. (1986). A non-classical logic for information retrieval. *The computer journal*, Vol. 29, No. 6, pp. 481–485.
33. Vintar, S. & Fiser, D. (2008). Harvesting multi-word expressions from parallel corpora. *LREC*.
34. Zhang, W., Yoshida, T., & Tang, X. (2008). Text classification based on multi-word with support vector machine. *Knowledge-Based Systems*, Vol. 21, No. 8, pp. 879–886.

Article received on 16/01/2019; accepted on 04/03/2019.  
Corresponding author is Rim Koulali.



# A Deep Attention based Framework for Image Caption Generation in Hindi Language

Rijul Dhir\*, Santosh Kumar Mishra\*, Sriparna Saha, Pushpak Bhattacharyya

Indian Institute of Technology Patna,  
India

{rizul.cs15, santosh\_1821cs03, sriparna, pb}@iitp.ac.in

**Abstract.** Image captioning refers to the process of generating a textual description for an image which defines the object and activity within the image. It is an intersection of computer vision and natural language processing where computer vision is used to understand the content of an image and language modelling from natural language processing is used to convert an image into words in the right order. A large number of works exist for generating image captioning in English language, but no work exists for generating image captioning in Hindi language. Hindi is the official language of India, and it is the fourth most-spoken language in the world, after Mandarin, Spanish and English. The current paper attempts to bridge this gap. Here an attention-based novel architecture for generating image captioning in Hindi language is proposed. Convolution neural network is used as an encoder to extract features from an input image and gated recurrent unit based neural network is used as a decoder to perform language modelling up to the word level. In between, we have used the attention mechanism which helps the decoder to look into the important portions of the image. In order to show the efficacy of the proposed model, we have first created a manually annotated image captioning training corpus in Hindi corresponding to popular MS COCO English dataset having around 80000 images. Experimental results show that our proposed model attains a BLEU1 score of 0.5706 on this data set.

**Keywords.** Image captioning, Hindi language, convolutional neural network, recurrent neural network, gated recurrent unit, attention mechanism.

---

\*Both authors have equally contributed.

## 1 Introduction

Modern visual classifiers can recognize many of the object categories [6, 12] which have been a main focus in the computer vision community. Automatic caption generation is one of the challenging tasks in artificial intelligence because it requires recognition of not only objects but also other visual elements such as actions, attributes as well as generating sentence which describes relationships between objects, actions, and attributes in the image [15]. It has emerged as a challenging and important research area because of the recent advancement in statistical language modeling and image recognition [1, 13]. Generating a meaningful language description of an image requires image understanding that is quite different than image classification and object recognition. This is one of the interesting problems in artificial intelligence because it connects computer vision with natural language processing.

Image captioning could have great impact by helping visually impaired people to better understand the content on the web. Image captioning came into existence with recent advancements in machine translation where the task is to transform a sentence  $S$  written in a source language into translated target language  $T$ , by maximizing  $p(T|S)$

Generating captioning of an image is very much similar to machine translation where the source language is pixels of an image, and the target language is a sentence. In this case, convolutional neural network (CNN) is used in place of RNN [15].

In recent works, it is shown that CNN can produce a rich representation of the input image by embedding it to a fixed length vector representation such that it can be used for various computer vision tasks. Hence, CNN can be used as an encoder, by first pre-training it for image classification task and then using the last hidden layer as an input to the RNN which can work as a decoder to generate the sentences (refer to Fig. 1).

Hindi is one of the oldest languages in the world which is spoken in India and South Asia. It has a direct line of evolution to Sanskrit [11]. It is the fourth most spoken language in the world. Hindi is part of one of the oldest religious and literary tradition in the world. India is a linguistically diverse country, Indian constitution has adopted 15 languages as national importance and Hindi and English as official languages. Most of the people in India use Hindi language for communication. Hence there is a need to develop systems for Hindi language so that most of the people can get the benefit of recent research works.

Currently, Government of India is promoting Hindi language in all its departments and organizations by celebrating Hindi Diwas every year. Government is motivating its employees to work in Hindi in place of English. This work can be helpful to all public organizations where captions of images in Hindi are needed. We did not find any previous work on Hindi language image captioning. Inspired by this, in the current work, we aim to develop attention based Hindi language image captioning framework utilizing Resnet 101 [6] as Encoder and GRU based deep-learning model [3] as a decoder with attention.

To the best of our knowledge, this is the first work where an image captioning framework is developed for Hindi language. Firstly we have used a pre-trained model to interpret the content of images. There are several CNN architectures available, but we have used Resnet 101 as it was shown in the literature that it attained the best result and it uses the residual connections which can solve the degradation problem [6]. Resnet 101 architecture is used to extract features of the images by saving the output of the last hidden layer as we are interested in the internal representation of images instead of classification.

Here we have used Gated Recurrent Unit (GRU) [3] as a decoder. The output of the last hidden layer of Resnet 101 is passed to attention, and the context vector generated by it is used as an input to GRU which generates a description for the image. This architecture yields better performance as compared to standard CNN and RNN architectures where Long Short Term Memory (LSTM) is used as a decoder.

As no Hindi corpus was available for solving the task of image captioning, in a part of the work we have manually created a Hindi corpus. MS COCO [9] is a publicly available English image captioning data set containing images. This is large image dataset which is used for object detection, image segmentation, caption generation etc. We have used this dataset for image captioning. In this dataset, each image has five captions. We have manually translated this corpus for Hindi language. The proposed approach is applied to this manually annotated MS COCO Hindi dataset to show model's efficiency and to prove the supremacy of the proposed approach. We have obtained the BLEU-1 score of 0.5706, BLEU-2 score of 0.3914, BLEU-3 score of 0.2935 and BLEU-4 score of 0.1726. In order to establish the efficacy of Resnet 101 as encoder and GRU as the decoder for the task of Hindi caption generation, we have also reported the results of many other combinations of encoders and decoders utilizing models of Inception v4 [14], GRU [3] and LSTM [7]. Results demonstrate the superiority of the proposed architecture.

## 2 Related Work

In the literature, there exist two types of approaches for generating image captioning, the first one is the top-down approach [1][13][17] and the second one is the bottom-up approach [5][8][4].

The top-down approach starts from the input image and converts it into words while the bottom up approach first comes up with words which describe the various aspects of an image and then combines them to generate the description. In the top-down approach, there is an end to end formulation from an image to sentence using the recurrent neural network, and all the parameters



of the recurrent neural network are learned from training data. Limitations of the top-down approach are that sometimes fine details can not be obtained which may be important to generate the description of an image. Bottom-up approaches do not have this problem as they can be operated on any image resolution, but there is a lack of end to end formulation which is going from image to sentence.

Bottom-up approach starts with visual concepts, objects, attributes, words and phrases which are combined using the language models. Farhadi et al. [5] defined a method that can compute a score linking to an image. This score is used to attach a description to an image.

Top-down approaches are the recent ones; in these approaches, image captioning problem is formulated as a machine translation problem. In machine translation, one language is translated to another one, but in case of image captioning, the visual representation is translated to language. Sutskever et al. [13] proposed a sequence end to end approach for sequence learning. They have used a multilayered Long Short-Term Memory (LSTM) to encode the input sequence to a vector of fixed dimensionality, and then another deep LSTM is used to decode the target sequence from the vector.

Bahdanau et al. [1] developed the encoder-decoder architecture for machine translation by allowing a model to automatically search the part of the source sentence that is relevant in predicting the target word. To the best of our knowledge, there does not exist any image captioning model for Hindi language. In this work, we have proposed an attention based Encoder-Decoder model for Hindi language captioning. Also, we have used GRU instead of LSTM due to its several advantages.

### 3 Proposed Methodology

In this paper, we have developed a method of caption generation based on neural and probabilistic framework. Recent development in statistical machine translation has provided powerful sequence models to generate state-of-the-art results by maximizing the probability of correct translation given an input sequence in an end to end approach.

Thus, we use the probabilistic framework to directly maximize the probability of correct description given an image by using the following formulations:

$$\theta^* = \arg \max_{\theta} \sum_{(I,S)} \log p(S|I; \theta), \quad (1)$$

where  $I$  is an image,  $\theta$  is the parameter of the model and  $S$  is the generated description. Since  $S$  is the generated description, its length should be unbounded. Therefore we can apply the chain rule to find the joint probability over  $S_0, \dots, S_N$ , where  $N$  is the assumed length of the description:

$$\log p(S|I) = \sum_{i=0}^N \log p(S_i|I, S_0, S_1, \dots, S_{i-1}), \quad (2)$$

here  $\theta$  is omitted for convenience.  $(S, I)$  is ordered pair from training example where  $I$  is an image and  $S$  is the corresponding description. Our aim is to optimize the sum of log probabilities as described in (2) over the whole training set using Adaptive Moment Estimation (Adam) optimizer.

Here  $p(S_i|I, S_0, S_1, \dots, S_{i-1})$  is modeled using a Recurrent Neural Network (RNN); here variable number of words is conditioned upon  $i-1$  expressed by hidden state of fixed length,  $h_i$ . This hidden memory is updated after scanning a new input,  $x_i$ , by using a non-linear function,  $f$ :

$$h_{i+1} = f(h_i, x_i), \quad (3)$$

here Gated Recurrent Unit (GRU) is used as function  $f$  which has shown state of the art performance on the sequence to sequence translation.

#### 3.1 Convolutional Neural Network

Convolutional Neural Network (CNN) is used for the internal representation of the image. CNN has been widely used for object recognition and object detection task in computer vision. We have used Resnet 101 convolution neural network architecture given by Kaiming He et al. [6] which won the 1st place in the ILSVRC 2015 classification competition.

This is a 101 layer CNN model pre-trained on ImageNet dataset. We have removed the last layer of the model as it is used for classification but we are more interested in the internal representation of images. We have pre-processed images with the Resnet 101 model and have extracted features from block4 layer as input for further processing.

The extractor produces  $L$  vectors, each of which is a  $D$ -dimensional representation corresponding to a part of the image:

$$a = \{a_1, \dots, a_L\}, a_i \in \mathbb{R}^D. \quad (4)$$

Global image feature can be obtained by:

$$a_g = \frac{1}{L} \sum a_i. \quad (5)$$

Image Vector and Global Image Vector can be obtained by using a single layer perceptron with rectifier activation function:

$$v_i = \text{ReLU}(\mathbf{W}_a a_i), \quad (6)$$

$$v_g = \text{ReLU}(\mathbf{W}_g a_g), \quad (7)$$

where  $\mathbf{W}_a$  and  $\mathbf{W}_g$  are the weight parameters,  $L$  is number of vectors and  $D$  is size of each vector. The transformed spatial image feature form is  $V = [v_1, \dots, v_L]$ .

### 3.2 Attention

Attention is a process of concentrating on a discrete aspect of information while ignoring other perceivable information [2]. It is a way to instruct the model where to concentrate to generate the corresponding word rather than the whole image. At time  $t$ , based on the hidden state, the decoder would attend to the specific regions of the image and compute context vector using the spatial image features from a convolution layer of a CNN:

$$c_t = g(V, h_t). \quad (8)$$

We feed  $V$  and  $h_t$  through a single layer neural network followed by a softmax function to generate the attention distribution over the  $k$  regions of the image:

$$z_t = \mathbf{W}_h^T \tanh(\mathbf{W}_v V + (\mathbf{W}_g h_t) \mathbf{1}^T), \quad (9)$$

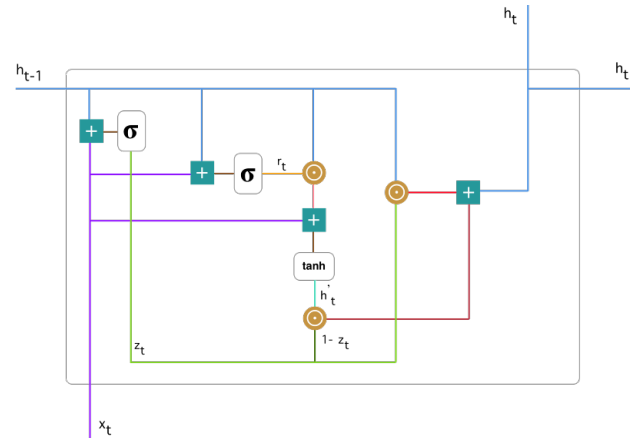


Fig. 1. Gated Recurrent Unit

$$\alpha_t = \text{softmax}(Z_t), \quad (10)$$

where  $\mathbf{1}^k$  is a vector with all elements set to 1.  $\mathbf{W}_v, \mathbf{W}_g \in \mathbb{R}^{L \times D}$  and  $\mathbf{W}_h \in \mathbb{R}^L$  are parameters to be learnt.  $\alpha \in \mathbb{R}^L$  is the attention weight over features in  $V$ . Based on the attention distribution, the context vector  $c_t$  can be obtained by:

$$c_t = \sum (\alpha_{ti} V_{ti}). \quad (11)$$

### 3.3 Gated Recurrent Unit (GRU) based Sentence Generator

$x_t$  is the input vector and we obtain it by concatenating the word embedding vector,  $w_t$ , global image feature vector,  $v_g$ , and context vector,  $c_t$ , to get the input vector.

$$x_t = [w_t; v_g; c_t]. \quad (12)$$

GRU was introduced by Kyunghyun Cho et al. [3] and applied successfully for machine translation and sequence generation. GRU is an improved version of the Recurrent Neural Network. To solve the vanishing gradient problem of standard RNN, it uses update gate and reset gate. The behaviour of the cell is controlled by these two gates.

The core of the GRU model is memory cell which stores knowledge of every time step (what input has been observed up to present state (see

Figure 1<sup>1</sup>)). Basically, the update gate and reset gate are the vectors responsible for deciding which information should be passed to the output. These two gates are trained to store the information from the previous time steps without wasting those as well as removing the pieces of information which are irrelevant for the prediction.

Our proposed image captioning architecture can be divided into the following components as shown in Figure 2:

- RESNET 101 Convolutional Neural Network which is pre-trained on the ImageNet dataset, is used to extract features from the images. Here CNN is working as feature extractor, or it can work as an encoder which converts images into fixed length vector representations. In our case, it is converted to feature vector of dimension  $49 \times 2048$ .
- We have used a function to summarize the image and get global image vector,  $v_g$  which is passed to GRU for global information about image.
- Attention mechanism is used to obtain context vector,  $c_t$  which tells the model where to look in the image for corresponding word generation.
- There is a word embedding layer for handling the text input; it provides a dense representation of the words and their relative meanings. It is used to fit a neural network on the text data. Word-embedding vector  $w_t$  is given as input to the GRU.
- Input to GRU is Context vector, Word Embedding vector and global image vector with previous hidden state and it will predict the hidden state,  $h_t$ , based on these; here we have used GRU layer with 512 neurons that will provide a vector having 512 elements to represent a word.
- Both the context vector and the hidden state are passed to Multi-Layer Perceptron (MLP) Network to make predictions over the entire vocabulary for the next word for the caption.

<sup>1</sup><https://towardsdatascience.com/understanding-gru-networks-2ef37df6c9be>

Here CNN is working as feature extractor, generating features of dimension  $49 \times 2048$ ; it is further processed by attention mechanism to produce a 2048 representation of the area of an image to look into. Global vector provides an image summary for GRU. Word Embedding provides a dense representation of the words and their relative meanings which is input to GRU. The outputs of the GRU layer and attention mechanism are then fed to MLP network which makes the prediction over the entire vocabulary for the next word in the sequence (see Fig 2).

### 3.4 Hyparameters of the Model

The input to the Resnet 101 convolutional neural network is  $256 \times 256$  RGB images. The features are extracted from **block4** layer of the neural network with size  $49 \times 2048$ . The input captions are fed into an embedding layer with 512 neurons. We have used 0.5 dropout to avoid over-fitting. Here we have used softmax cross-entropy to get the loss and Adam optimizer to optimize our loss with a learning rate of  $4e - 4$  and batch size of 32. To train the proposed model with 80000 images on Cluster of 8 GPU Nvidia GTX 1080, approximately 14 hours were needed and to test 8000 images; it takes approximately 15 minutes to generate the description of the input images.

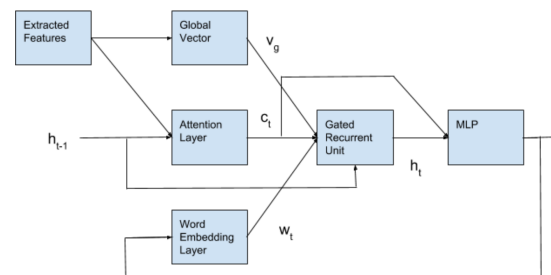


Fig. 2. Process flow diagram of proposed method

## 4 Experimental Setup

In this section, we have described the procedure of data set creation and experimental results.

#### 4.1 Dataset Creation

We have chosen MS COCO dataset [9] for our experiment which is a commonly used data set for caption generation in English language. This data set includes complex everyday scenes with common objects in naturally occurring contexts and can be downloaded easily. Hence, it covers large possible categories of images.

As no Hindi image captioning data set is available in the literature, we have manually annotated the captions of this MS COCO data set to generate a gold standard image captioning data set for Hindi language as shown on Figure 3 and Figure 4. Firstly Google translator is applied to convert English captions to Hindi captions. We then had two annotators with the inter-annotator agreement of 84% to manually check and correct the translation on sample data of 400 captions. These annotators verify and correct the outputs obtained from Google Translator. Through this process, we convert MS COCO English dataset to Hindi language dataset.

The COCO dataset is an excellent object detection dataset with 80 classes, 80,000 training images and 40,000 validation images and each of the images is paired with five different captions which provide a clear description of the salient entities and events. Validation set contains 8000 images to monitor the performance of the trained model. When the performance of a trained model improves at the end of any epoch, then we save that model. At last, we can get the best-learned model on the training dataset. The dataset contains a test set which has 8000 images. We evaluated the performance of the learned model and its prediction on a test set.

#### 4.2 Preprocessing

We truncate captions longer than 18 words present in images of COCO dataset. We then build a vocabulary of words that occur at least three times in the training set, resulting in 9034 words for COCO.



**Fig. 3.** Example Image for Dataset Preparation wrt Fig. 4

English Captions	Google translated Captions in Hindi	Corrected Captions in Hindi
closeup of bins of food that include broccoli and bread	भोजन के डिब्बे को बंद करना जिसमें ब्रोकोली और ब्रेड शामिल हैं	भोजन के वो डिब्बे जिनमें ब्रोकोली और ब्रेड शामिल हैं उनकी पास की तस्वीर
a meal is presented in brightly colored plastic trays	एक भोजन चमकीले रंग की प्लास्टिक ट्रे में प्रस्तुत किया जाता है	भोजन चमकीले रंग की प्लास्टिक ट्रे में प्रस्तुत किया जाता है
there are containers filled with different kinds of foods	विभिन्न प्रकार के खाद्य पदार्थों से भरे कंटेनर हैं	विभिन्न प्रकार के खाद्य पदार्थों से भरे कंटेनर हैं
a bunch of trays that have different food	ट्रे का एक गुच्छा जिसमें अलग भोजन होता है	ट्रे का एक समूह जिसमें अलग अलग भोजन है
colorful dishes holding meat vegetables fruit and bread	मांस सब्जी फल और रोटी पकड़े रंगीन व्यंजन	रंगीन व्यंजन जिनमें मांस सब्जी फल और रोटी है

**Fig. 4.** Example of Dataset Preparation

#### 4.3 Chosen Technique for Comparative Analysis

BLEU (Bilingual Evaluation Understudy Score) is a popular evaluation technique to check the similarity of a generated sentence with respect to a reference sentence. BLEU score evaluation technique was proposed by Papineni et al. [10] in the year 2002. In the current work, this metric is used to evaluate the performance of our caption generation model.

### 5 Results and Discussions

This section will cover the quantitative and qualitative analysis of the generated descriptions of the images. During our research, we could not find any previous paper related to Hindi language image caption generation. So this is the first work in this field to the best of our knowledge. In order to compare the performance

of the proposed architecture we have proposed the following baselines:

- Baseline 1: In this baseline, Resnet 101 is used as an encoder and LSTM is used as a decoder.
- Baseline 2: In this baseline, Inception v4 is used as an encoder and LSTM is used as a decoder.
- Baseline 3: In this baseline, Inception v4 is used as an encoder and GRU is used as a decoder.

Our proposed model is data-driven, and it is trained end to end. We have used manually annotated MS COCO Hindi dataset for training purpose. Hence proposed architectures can be trained on CPU and GPU. We have tested the performance of our proposed model on MS COCO test dataset.

### 5.1 Qualitative Analysis

Results of generated descriptions on the test images are given in Figure 5. As can be seen from the generated descriptions, these captions are pretty much close to the input images but still there is a need for improvement in the generated descriptions and we are working towards that. We can improve the quality of generated description for an image by training using larger dataset.

### 5.2 Quantitative Analysis

Although it is possible to evaluate the quality of generated descriptions of images manually, sometimes it is not very clear that generated description is successful or not given an image. Hence it is necessary to have a subjective score that decides the usefulness of each description given an image. There are many evaluation metrics in the literature which are used in machine translation. The most commonly used metric in machine translation literature is BLEU score [10].

Here we have used the BLEU score for evaluation of generated description of an image. In the dataset, each image has been annotated by five sentences that are relatively visual and

unbiased. There are 8000 images in the test set of this dataset to test the performance of the proposed model. BLEU has been recorded for the test dataset. We have evaluated BLEU-1, BLEU-2 BLEU-3 and BLEU 4 scores of our model and all the three baselines which are given in Table-1. Results show that our proposed model outperforms all the baselines. The proposed method has obtained BLEU-1 score of 57.0, BLEU-2 score of 39.4, BLEU-3 score of 26.4 and BLEU-4 score of 17.3, which are significantly better than other three baselines as shown in Table-1.

### 5.3 Statistical Test

Welch's t-test, [16] statistical hypothesis test, has been conducted at 5%(0.05) significance level to show that performance improvements attained by our proposed method over other baselines are statistically significant, not happened by chance. BLEU scores are obtained by 10 consecutive runs of each architecture. Smaller p-values indicate better performance of the proposed architecture compared to other architectures. The p-values obtained by the statistical test are reported in Table- 2. Results obtained established the statistical significance of the improvements obtained.

### 5.4 Error Analysis

In this section, we have tried to analyze the errors made by our proposed system. In Fig-5 (a) generated description reports the action of 'sitting on the tree' in place of 'sitting on a bear', This may happen because there are many images in training data where the bear is associated with the tree. More frequent words in the vocabulary have more probabilities during softmax prediction. This is the main reason for an error in the generated description.

In Fig-5 (b), generated descriptions predicted 'cat' in place of 'dog' because the head of the dog is not visible. In Fig-5 (c), the generated caption is very close to activity and object inside the image, but the terms 'glass' and 'bowl' are missed in the generated description. In Fig-5 (d), the model is not able to differentiate between the terms 'sofa' and 'chair'.



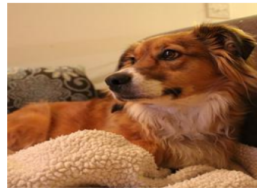
(a) एक भूरा भालू एक पेड़ के ऊपर बैठा है ।



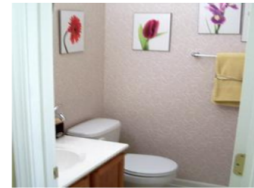
(b) एक बिल्ली एक बिस्तर पर सो रही है ।



(c) एक सफेद प्लेट जिसमें ब्रोकोली और मांस शामिल हैं ।



(d) एक कुत्ता एक सोफे पर एक कुर्सी पर बैठा है ।



(e) एक बाथरूम में एक शौचालय और एक सिंक है ।



(f) एक मेज पर बैठे लोगों का एक समूह ।

**Fig. 5.** Some Examples of Generated Captions by the Proposed Model

**Table 1.** BLEU score of different architectures

State of Arts	BLEU-1 Score	BLEU-2 Score	BLEU-3 Score	BLEU-4 Score
<b>Resnet 101 and GRU</b>	<b>57.0</b>	<b>39.1</b>	<b>26.4</b>	<b>17.3</b>
Baseline 1	56.4	38.8	26.3	17.2
Baseline 2	56.3	38.4	25.8	16.9
Baseline 3	55.7	38.4	25.9	16.8

**Table 2.** p-values produced by Welch's t-test comparing our proposed method with other baselines

State of Arts	BLEU-1 Score	BLEU-2 Score	BLEU-3 Score	BLEU-4 Score
Baseline 2	6.86249e-34	1.58517e-23	2.68755e-36	1.67851e-10
Baseline 2	2.68755e-36	2.68755e-36	6.86249e-34	1.0313e-27
Baseline 3	3.67333e-46	2.68755e-36	1.82938e-29	4.42441e-31

In Fig-5 (e), the model has correctly predicted objects of the input image. For the description of Fig-5 (f), the model has predicted people sat on the 'table' in place of 'chair'; this is because 'chair' is not visible in the input image. Below we have enumerated the possible reasons for getting errors:

- Most of the errors occurred due to non-presence of certain words in the training captions. Increasing the size and diversity of the training data may help in reducing these types of errors.

- Most of the errors occurred because certain phrases are frequently occurring in the training data. This can be solved by the use of a suitable unbiased training set.

## 6 Conclusions and Future Work

We have presented an attention-based image captioning model trained end to end that can view an image and generate a description in the Hindi language. The proposed model is based on a convolution neural network to encode an image into a fixed length vector representation which is



further used by attention layer to produce context vector which is further used by the recurrent neural network. As there was no previous work done for Hindi language Image Captioning, we have tried various combinations for encoder and decoder and treated them as baseline models. The best result we attained is from a combination of Resnet 101 as an encoder and GRU as a decoder. Although GRU and LSTM provide similar results, GRU is more efficient due to its less complex structure and fewer number of gates. In future, we will explore the possibility of an ensemble of various possible encoder and decoder architectures to improve the result further. We will also try different sampling techniques to remove the bias towards certain phrases.

## References

1. Bahdanau, D., Cho, K., & Bengio, Y. (2014). Neural machine translation by jointly learning to align and translate. *arXiv preprint arXiv:1409.0473*.
2. Cho, K., Courville, A., & Bengio, Y. (2015). Describing multimedia content using attention-based encoder-decoder networks. *IEEE Transactions on Multimedia*, Vol. 17, No. 11, pp. 1875–1886.
3. Cho, K., Van Merriënboer, B., Gulcehre, C., Bahdanau, D., Bougares, F., Schwenk, H., & Bengio, Y. (2014). Learning phrase representations using rnn encoder-decoder for statistical machine translation. *arXiv preprint arXiv:1406.1078*.
4. Elliott, D. & Keller, F. (2013). Image description using visual dependency representations. *Proceedings of the 2013 Conference on Empirical Methods in Natural Language Processing*, pp. 1292–1302.
5. Farhadi, A., Hejrati, M., Sadeghi, M. A., Young, P., Rashtchian, C., Hockenmaier, J., & Forsyth, D. (2010). Every picture tells a story: Generating sentences from images. *European conference on computer vision*, Springer, pp. 15–29.
6. He, K., Zhang, X., Ren, S., & Sun, J. (2016). Deep residual learning for image recognition. *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 770–778.
7. Hochreiter, S. & Schmidhuber, J. (1997). Long short-term memory. *Neural computation*, Vol. 9, No. 8, pp. 1735–1780.
8. Kulkarni, G., Premraj, V., Dhar, S., Li, S., Choi, Y., Berg, A. C., & Berg, T. L. (2011). Baby talk: Understanding and generating image descriptions. *Proceedings of the 24th CVPR*, Citeseer.
9. Lin, T.-Y., Maire, M., Belongie, S., Hays, J., Perona, P., Ramanan, D., Dollár, P., & Zitnick, C. L. (2014). Microsoft coco: Common objects in context. *European conference on computer vision*, Springer, pp. 740–755.
10. Papineni, K., Roukos, S., Ward, T., & Zhu, W.-J. (2002). Bleu: a method for automatic evaluation of machine translation. *Proceedings of the 40th annual meeting on association for computational linguistics*, Association for Computational Linguistics, pp. 311–318.
11. Rubino, C. R. G., Garry, J., & Rubino, C. (2001). *Facts about the world's languages: An encyclopedia of the world's major languages, past and present*. Hw Wilson.
12. Simonyan, K. & Zisserman, A. (2014). Very deep convolutional networks for large-scale image recognition. *arXiv preprint arXiv:1409.1556*.
13. Sutskever, I., Vinyals, O., & Le, Q. V. (2014). Sequence to sequence learning with neural networks. *Advances in neural information processing systems*, pp. 3104–3112.
14. Szegedy, C., Ioffe, S., Vanhoucke, V., & Alemi, A. A. (2017). Inception-v4, inception-resnet and the impact of residual connections on learning. *AAAI*, volume 4, pp. 12.
15. Vinyals, O., Toshev, A., Bengio, S., & Erhan, D. (2015). Show and tell: A neural image caption generator. *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 3156–3164.
16. Welch, B. L. (1947). The generalization of student's problem when several different population variances are involved. *Biometrika*, Vol. 34, No. 1/2, pp. 28–35.
17. Wu, Q., Shen, C., Liu, L., Dick, A., & van den Hengel, A. (2016). What value do explicit high level concepts have in vision to language problems? *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 203–212.

Article received on 23/01/2019; accepted on 04/03/2019.  
Corresponding author is Rijul Dhir.





# An Ensemble of Automatic Keyword Extractors: TextRank, RAKE and TAKE

Tayfun Pay<sup>1</sup>, Stephen Lucci<sup>2</sup>, James L. Cox<sup>3</sup>

<sup>1,3</sup> Computer Science Department,  
Graduate Center of New York, New York,  
United States

<sup>2</sup> Computer Science Department,  
The City College of New York, New York,  
United States

<sup>3</sup> Brooklyn College of New York,  
Computer and Information Science Department, Brooklyn,  
United States

tpay@gradcenter.cuny.edu, lucci.stephen@gmail.com, cox@sci.brooklyn.cuny.edu

**Abstract.** We construct an ensemble method for automatic keyword extraction from single documents. We utilize three different unsupervised automatic keyword extractors in building our ensemble method. These three approaches provide candidate keywords for the ensemble method without using their respective threshold functions. The ensemble method combines these candidate keywords and recomputes their scores after applying pruning heuristics. It then extracts keywords by employing dynamic threshold functions. We analyze the performance of our ensemble method by using all parts of the Inspect data set. Our ensemble method achieved a better overall performance when compared to the automatic keyword extractors that were used in its development as well as to some recent automatic keyword extraction methods.

**Keywords.** Data mining, text mining, text analysis, ensemble methods.

## 1 Introduction

This paper is an expansion of the work that was originally presented in [22]. This extended version includes a thorough explanation of the automatic keyword extractors that are used in building our ensemble method. We also provide a sample input

and output for our ensemble method as well as the methods used in its construction. We present the performance of our ensemble method on additional data sets. We also compare its performance with some recent automatic keyword extractors.

There has been enormous amount of data that has been generated in recent years and there is a need to process this data for different purposes. Simultaneously, there has been a growing interest in designing keyword<sup>1</sup> extractors that unearth words or sequence of words that concisely represent a document.

These automatic keyword extraction approaches could be graph based [14], statistics based [18], clustering based [16], linguistic based [11], or other [10]. They could also be some combination of the aforementioned methods as in [20, 5, 12]. There has been also various studies done using semi supervised [1] as well as supervised [15] approaches.

Extracting more significant keywords is important for many different tasks in big data such

---

<sup>1</sup>We use the words, keyword and keyphrase interchangeably although some authors refer to the former as having a single word and the latter as having more than one word.

as classification [7], clustering [28], indexing [9] and data-analysis [6]. For instance, when more significant keywords are extracted then the subsequently utilized classification algorithms could potentially place the documents into more relevant categories. Similarly, more significant keywords could conceivably assist the clustering algorithms to create more appropriate clusters with the given documents. Additionally, when more significant keywords are extracted then they could possibly be used to decide on the placement of the document within the given database more accurately. There is basically no down side to extracting more significant keywords.

There has also been growing interest in ensemble based machine learning methods, such as the ones in [24, 29, 2]. An ensemble based machine learning approach combines several machine learning methods, wherein each one is ran independently with the same input and the output is then agreed upon by merging their outcomes. If a simple majority opinion is obtained among these approaches, it is referred to as hard voting. Instead, if each approach assigns a probability of class membership, then the average probability obtained is referred to as soft voting. It has been recognized that the performance of ensemble based machine learning methods are often better than any single approach used in their construction. This is particularly true when these machine learning approaches are autonomous from each other, such that they make uncorrelated errors.

In this light, we decided to construct an ensemble method for automatic keyword extraction. We employ the following unsupervised automatic keyword extractors in the construction of our ensemble method: TextRank [19], RAKE [23] and TAKE [21]. In the following section, we discuss how these automatic keyword extractors work. We then explain how to construct our ensemble method.

In the two sections that follow, we introduce the Inspect data set from [13] and our metrics for measuring the performance of our approaches; and go through a sample input and output to the three automatic keyword extractors as well as to our ensemble method. We then analyze the performance of our ensemble method and

compare it to the methods used in its construction as well as to other automatic keyword extractors. Finally, we conclude with prospects for our current and future work on this topic.

## 2 Automatic Keyword Extractors

### 2.1 Text-Rank

Text-Rank [19] utilizes a part of speech (pos) tagger to assign a pos-tag for each word. It only considers adjectives and nouns as possible keyword components. These words are treated as vertices in a graph wherein an edge is drawn between each word that appears within a given co-occurrence window. A co-occurrence window of size  $n$  for a given word consists of the  $n - 1$  words that appear to the left and to the right of it. Then the ranking algorithm, page-rank [4], is executed until convergence is reached. The top third scoring words are selected for post-processing. At this stage, multi-word keywords are constructed by consulting the original text to determine if any of the selected words appear next to each other. Any word that does not appear next to another word in the list of the top third scoring words of the document are extracted as single-word keywords, and the rest are extracted as multi-word keywords.

### 2.2 RAKE

RAKE (Rapid Automatic Keyword Extraction) [23] utilizes a stop-list to locate candidate keywords. Any sequence of words that appear between two stop-list words and/or punctuation marks are marked as candidate keywords. Then the frequency and the degree values of each word in the list of candidate keywords are calculated. The frequency of a word is the total number of its occurrences within the list of candidate keywords. The degree of a word is the total number of words that it appears with, within the list of candidate keywords. Then each word is assigned a score of degree over frequency. The cumulative score of each candidate keyword is computed by summing up the scores of the words that it contains. The top third scoring candidate keywords are extracted as keywords.

## 2.3 TAKE

TAKE (Totally Automated Keyword Extraction) [21] utilizes a pos-tagger to assign a pos-tag for each word and then marks all noun-phrases as candidate keywords. A noun-phrase consists of zero or more adjectives followed by one or more nouns. Then all candidate keywords are filtered out in the following manner: 1) if they contain a word from a stop-list and 2) if they contain only one word with a frequency of one that does not appear in the first ten-percent of the document. TAKE calculates the candidate keyword scores in the same manner as RAKE. All candidate keywords that have a score higher than the value computed by the dynamic threshold function are extracted as keywords.

## 3 Ensemble of Automatic Keyword Extractors

There are several utility functions that need to be provided and several parameters that need to be set for the automatic keyword extractors in question to execute. First of all, RAKE and TAKE need access to a stop-list, the former needs a stop-list to be able to come up with candidate keywords and the latter needs one to filter the list of candidate keywords.

We use the fox stop-list from [8] for the aforementioned purposes. Second of all, TAKE and TextRank need a pos-tag for each word to be able to construct a set of candidate keywords. We use the default pos-tagger from the NLTK library [3] for this purpose. Finally, we set the co-occurrence window in TextRank to two.

Our ensemble method consists of the following four stages: A) Receiving a list of candidate keywords from each automatic keyword extractor. B) Filtering the list of candidate keywords according to pruning heuristics. C) Combining and recalculating the scores of candidate keywords. D) Applying a dynamic threshold function to extract keywords.

## 3.1 Candidate Keyword Selection

We remove the threshold function of each automatic keyword extractor. In doing so, the ensemble method gets the total set of candidate keywords found by each approach. The scores of each candidate keyword for each approach are normalized by dividing them by the highest score within their set. This provides the ensemble method with a list of candidate keywords for each approach with a possible score that is greater than zero and less than or equal to one.

There are various reasons why we remove the threshold function of each automatic keyword extractor. In one scenario, some candidate keywords that are discarded, due to scoring lower than the threshold function, might be found by another approach. In another extreme scenario, the normalized score of some candidate keyword might actually be higher in one approach than when compared to another. However, the one with the higher normalized score might not be selected due to scoring lower than the value set by the respective threshold function. Therefore, we take into account all of the candidate keywords from all of the automatic keyword extractors.

## 3.2 Pruning Heuristics

The ensemble method then applies a pruning heuristic. This heuristic removes any candidate keyword that is located by a single approach and consists of a single word. The rationale here is that there is no statistical significance for a keyword that consists of a single word and was only found by a single keyword extractor.

## 3.3 Recomputation of Candidate Keyword Scores

The next step is to recompute the candidate keyword scores. As we combine the three different lists of candidate keywords, the scores of candidate keywords located by more than one approach are summed and then multiplied by the total number of approaches that found them.

The rationale behind multiplying the summed scores by the total number of approaches is that their cumulative score might still be too low to pass

the threshold function in the next stage. This can happen even when a candidate keyword has been located by all of the approaches. Therefore, we want to additionally reward the candidate keywords that were found by more than one approach.

### 3.4 Dynamic Threshold Function

The keywords are finally extracted by applying dynamic threshold functions. In one method, the overall mean is calculated and any candidate keyword that scores higher than the mean is extracted as a keyword for the document. And in the other method, the overall median is utilized for this same purpose.

The dynamic threshold functions do not undermine the contextual properties of each document. This is because they do not treat them in the same way as a static threshold function would have. For instance, if a static threshold function was used, such as extracting top third scoring candidate keywords as keywords, then documents with the same number of candidate keywords would always have the same number of keywords extracted. On the other hand, we let the characteristics of the document determine how many keywords are extracted by employing dynamic threshold functions.

## 4 Data Set

The data set that we used was introduced in [13], and subsequently used in [19, 23, 21] for evaluating their automatic keyword extraction methods. This data set contains 2000 titles and abstracts for journal papers from *Computer Science and Information Technology*. These items are divided into a training set, validation set and testing set that contain 1000 and two sets of 500 documents, respectively. Only the testing set was used in [22] as well as in [19, 23, 21] since these are unsupervised methods. However, we also used the validation set and the training set for the purposes of testing. We will refer to the testing set, the validation set and the training set as data sets I, II and III, respectively.

Data set I contains 500 documents where there are 4912 manually assigned keywords of which

only 3837 are present in the titles and abstracts. Data set II contains 500 documents where there are 4575 manually assigned keywords of which only 3509 are present in the titles and abstracts. Finally, Data set III contains 1000 documents where there are 9788 manually assigned keywords of which only 7552 are present in the titles and abstracts.

We calculated the following parameters: extracted keywords, correct keywords, precision, recall and f-measure:

$$Precision = \frac{\text{correct keywords}}{\text{extracted keywords}},$$

$$Recall = \frac{\text{correct keywords}}{\text{manually assigned keywords}},$$

$$F - Measure = 2 \times \frac{\text{precision} \times \text{recall}}{\text{precision} + \text{recall}}.$$

Precision provides us with the percentage of extracted keywords that are correct. Recall provides the percentage of manually assigned keywords that are extracted. As previously noted in [13], both precision and recall are equally important so that they are given the same weight.

We use the total number of manually assigned keywords in the calculation of recall and f-measure, since it was done this way in [19, 23, 21]. Therefore, the highest obtainable recall for data set I is 78.1, for data set II is 76.7 and for data set III is 77.2.

## 5 Sample Input - Output

Table 1 illustrates the keywords that were extracted by the aforementioned methods and our ensemble approach from the following sample input, which is the title and the abstract of the paper in [25] and is from the Inspect data set [13]. The manually assigned keywords that are present in the title and the abstract are italicized.

“Title: An *optimization* approach to plan for *reusable software components*.

Abstract: It is well acknowledged in *software engineering* that there is a great potential for accomplishing significant *productivity improvements* through the implementation of a successful

*software reuse program*. On the other hand, such gains are attainable only by instituting detailed *action plans* at both the organizational and program level. Given this need, the paucity of research papers related to planning, and in particular, *optimized planning* is surprising. This research, which is aimed at this gap, brings out an application of *optimization* for the planning of *reusable software components* (SCs). We present a model that selects a set of SCs that must be built, in order to lower development and *adaptation costs*. We also provide implications to *project management* based on *simulation*, an approach that has been adopted by other cost models in the *software engineering* literature. Such a prescriptive model does not exist in the literature.”

**Table 1.** M1 = TextRank, M2 = RAKE, M3= TAKE and EN = Ensemble Method. The boldfaced candidate keywords are manually assigned keywords, N = did not locate, C = located as candidate keyword, E = extracted as keyword

<i>CandidateKeywords</i>	M1	M2	M3	EN
successful software reuse program	E	E	E	E
<b>software reuse program</b>	N	N	N	N
software engineering literature	E	E	E	E
<b>software engineering</b>	E	E	E	E
<b>productivity improvements</b>	E	N	E	E
significant productivity improvements	N	E	N	C
optimization approach	E	C	C	E
<b>optimization</b>	C	C	E	E
approach	C	C	C	E
detailed action plans	E	N	N	C
<b>action plans</b>	N	E	E	E
<b>optimized planning</b>	C	C	C	E
<b>reusable software components</b>	E	E	E	E
<b>adaptation costs</b>	E	C	C	E
<b>development costs</b>	N	N	N	N
lower development	N	C	N	C
development	C	N	C	C
project management based	N	E	N	C
<b>project management</b>	N	N	C	C
management	C	N	N	N
<b>simulation</b>	C	C	N	C
program level	C	E	E	E
prescriptive model	E	E	C	E
research papers related	N	E	N	C
research papers	E	N	E	E
organizational	E	C	N	C

In this example, there are 11 manually assigned keywords of which 10 of them are present in the text. (The manually assigned keyword *adaptation costs* is not present in the text as a whole.) TextRank extracted 11 keywords and 4 of them

were in the set of manually assigned keywords. RAKE extracted 10 keywords and 3 of them were in the set of manually assigned keywords. TAKE extracted 9 keywords and 5 of them were in the set of manually assigned keywords. The ensemble method extracted 14 keywords and 7 of them were in the set of manually assigned keywords.

It can be observed from table 1 that different automatic keyword extractors obtain different candidate keywords and extract varying keywords for the given document. The ensemble method was also able to extract keywords that were found only as candidate keywords by some automatic keyword extractor. For example, *optimized planning* was merely located as a candidate keyword by all of the automatic keyword extractors, but it was only extracted as a keyword by our ensemble approach. It is also interesting to see that none of the automatic keyword extractors were able to locate the keyword *software reuse program*, but instead located and extracted its superset as a keyword, *successful software reuse program*.

## 6 Analysis

The performance of our ensemble method along with the automatic keyword extractors that were used in its development on data sets I, II and III are illustrated in tables 2 and 3, 4 and 5, and 6 and 7, respectively.

For all three data sets, our ensemble method has the highest number of correct keywords extracted, highest recall as well as the highest f-measure compared to any of the individual automatic keyword extractors. This is true when either the mean or the median dynamic threshold function is utilized.

The only limitation is with respect to precision, where the method in [21] with the corresponding dynamic threshold function has a higher precision than our ensemble method, but a much lower recall. This is normal with respect to how ensemble based methods work. Because each approach contributed to the ensemble method certain keywords that were different from one another; and some of these keywords matched the manually assigned keywords and some did not.

In turn, this increased the recall, but brought down the precision at the same time. When we look at the results within each data set, we observe that our ensemble method achieved a higher precision with the mean dynamic threshold function compared to its iteration with the median dynamic threshold function. And it achieved a higher recall with the median dynamic threshold function compared to its iteration with the mean dynamic threshold function. This is also true with respect to the the method in [21]. It seems as though if you want higher precision then use the mean dynamic threshold function and if you want higher recall then use the median dynamic threshold function.

We also compare our ensemble method to the most recent automatic keyword extractor that was tested on the Inspec data set. The supervised recurrent neural network method in [26] achieves a precision of 31.0, recall of 27.5 and f-measure of 35.8 on data set I. This result is comparable to the performance of TextRank, but is inferior to the result achieved by our ensemble method.

There are some other methods, namely [16] and [27], that also employed the Inspec data set to test the performance of their automatic keyword extractors. The method in [16] calculated their recalls and consequently their f-measures, using the manually assigned keywords that are present in the abstracts. This provided them with an obtainable recall of 100. We cannot compare our methods to theirs in a fair manner since we cannot recompute their performance metrics given the limited information they provided. On the other hand, the authors of paper [27] created a subset of the Inspec data set by removing any document that does not contain all of its manually assigned keywords. This also provided them with an obtainable recall of 100. Once again, we cannot do a fair comparison between their methods and ours.

## 7 Conclusion

The quantity of data that is being collected is growing exponentially, and consequently, it becomes important that higher quality keywords be extracted. This is due to the fact that more

**Table 2.** Precision Recall and F-Measure for Data Set I

<i>method</i>	<i>precision</i>	<i>recall</i>	<i>f-measure</i>
Ensemble - (T=Mean) [22]	46.7	50.9	48.7
Ensemble - (T=Median) [22]	42.1	55.9	48.0
TAKE - (T=Mean) [21]	50.4	33.7	40.4
TAKE - (T=Median) [21]	44.3	46.9	45.6
RAKE - (ka-stoplist) [23]	33.7	41.5	37.2
RAKE - (fox-stoplist) [23]	26.0	42.2	32.1
TextRank (UnD. w=2) [19]	31.2	43.1	36.2
TextRank (UnD. w=3) [19]	28.2	38.6	32.6

**Table 3.** Extracted and correct keywords for Data Set I

<i>method</i>	<i>extracted</i>	<i>correct</i>
Ensemble - (T=Mean) [22]	5353	2501
Ensemble - (T=Median) [22]	6523	2746
TAKE - (T=Mean) [21]	3279	1653
TAKE - (T=Median) [21]	5197	2304
RAKE - (ka-stoplist) [23]	6052	2037
RAKE - (fox-stoplist) [23]	7893	2054
TextRank (UnD. w=2) [19]	6784	2116
TextRank (UnD. w=3) [19]	6715	1897

**Table 4.** Precision Recall and F-Measure for Data Set II

<i>method</i>	<i>precision</i>	<i>recall</i>	<i>f-measure</i>
Ensemble - (T=Mean)	44.0	49.1	46.7
Ensemble - (T=Median)	39.3	54.6	45.7
TAKE - (T=Mean)	45.1	34.5	39.1
TAKE - (T=Median)	40.0	46.2	42.9
RAKE - (fox-stoplist)	23.9	42.4	30.6
TextRank (UnD. w=2)	31.0	45.3	36.8

**Table 5.** Extracted and correct keywords for Data Set II

<i>method</i>	<i>extracted</i>	<i>correct</i>
Ensemble - (T=Mean)	5120	2248
Ensemble - (T=Median)	6357	2496
TAKE - (T=Mean)	3501	1578
TAKE - (T=Median)	5291	2114
RAKE - (fox-stoplist)	8124	1942
TextRank (UnD. w=2)	6681	2073

**Table 6.** Precision Recall and F-Measure for Data Set III

<i>method</i>	<i>precision</i>	<i>recall</i>	<i>f-measure</i>
Ensemble - (T=Mean)	43.8	50.2	46.9
Ensemble - (T=Median)	39.4	55.7	46.1
TAKE - (T=Mean)	46.1	35.4	40.1
TAKE - (T=Median)	40.7	48.4	44.2
RAKE - (fox-stoplist)	24.5	42.0	30.9
TextRank (UnD. w=2)	28.9	43.3	34.7

**Table 7.** Extracted and correct keywords for Data Set III

method	extracted	correct
Ensemble - (T=Mean)	11203	4918
Ensemble - (T=Median)	13835	5450
TAKE - (T=Mean)	7509	3464
TAKE - (T=Median)	11631	4734
RAKE - (fox-stoplist)	16748	4107
TextRank (UnD. w=2)	14631	4237

significant keywords would yield better results for the subsequently utilized machine learning algorithms. Additionally, this allows more accurate classification of documents and correct placement in databases.

We presented an unsupervised ensemble method for automatically extracting keywords from single documents. We showed that our ensemble method obtained better overall performance compared to the individual methods used in its development as well as to some of the recent approaches that were studied. We can also infer that our ensemble method is stable since it achieved this performance on three different data sets.

Although our ensemble method achieved better overall performance with respect to how well it found the keywords, it was nonetheless computationally slow because of TextRank. Our studies with regard to this has been presented in [17], where we construct yet another ensemble method without using TextRank and achieve comparable results, but with faster computation times. We wish to continue this line of research, where we will explore other types of ensemble methods for keyword extraction and experiment on different data sets.

## References

1. Aggarwal, A., Sharma, C., Jain, M., & Jain, A. (2018). Semi supervised graph based keyword extraction using lexical chains and centrality measures. *Computación y Sistemas*, Vol. 22, No. 4.
2. Alam, T., Ahmed, C. F., Zahin, S. A., Khan, M. A. H., & Islam, M. T. (2018). An effective ensemble method for multi-class classification and regression for imbalanced data. *Industrial Conference on Data Mining*, pp. 59–74.
3. Bird, S. & Loper, E. (2002). Nltk: the natural language toolkit. *ETMTNLP 02 Proceedings of the ACL-02.*, Vol. 1, pp. 63–70.
4. Brin, S. & Page, L. (1998). The anatomy of a large-scale hypertextual web search engine. *Computer Networks and ISDN Systems.*, Vol. 30, pp. 1–7.
5. Campos, R., Mangaravite, V., Pasquali, A., J., A. M., Nunes, C., & Jatowt, A. (2018). A text feature based automatic keyword extraction method for single documents. *European Conference on Information Retrieval*, pp. 684–691.
6. Castillo, E., Cervantes, O., & Vilarino, D. (2017). Text analysis using different graph-based representations. *Computación y Sistemas*, Vol. 21, No. 4, pp. 581–599.
7. Castro, D., Adame, Y., Pelaez, M., & Muñoz, R. (2017). Authorship verification, neighborhood-based classification. *Computación y Sistemas*, Vol. 21, No. 2.
8. Fox, C. (1989). A stop list for general text. *ACM SIGIR Forum*, Vol. 24, pp. 19–21.
9. Gelbukh, A., Sidorov, G., & Guzmán-Arenas, A. (2005). Document indexing with a concept hierarchy. *Computación y Sistemas*, Vol. 8, No. 4, pp. 281–292.
10. Giamblanco, N. & Siddavaatam, P. (2017). Keyword and keyphrase extraction using newton's law of universal gravitation. *Electrical and Computer Engineering (CCECE), 2017 IEEE 30th Canadian Conference on*, IEEE, pp. 1–4.
11. Hu, X. & Wu, B. (2006). Automatic keyword extraction using linguistic features. *Data Mining Workshops, 2006. ICDM Workshops 2006. Sixth IEEE International Conference on*, IEEE, pp. 19–23.
12. Huh, J. (2018). Big data analysis for personalized health activities: Machine learning processing for automatic keyword extraction approach. *Symmetry*, Vol. 10, pp. 93.
13. Hulth, A. (2003). Improved automatic keyword extraction given more linguistic knowledge. *Proceedings of the 2003 conference on empirical methods in natural language processing.*, pp. 216–223.
14. Litvak, M. & Last, M. (2008). Graph-based keyword extraction for single-document summarization. *Proceedings of the workshop on Multi-source Multilingual Information Extraction and Summarization*, Association for Computational Linguistics, pp. 17–24.

15. Liu, F., Liu, F., & Liu, Y. (2008). Automatic keyword extraction for the meeting corpus using supervised approach and bigram expansion. *Spoken Language Technology Workshop, 2008. SLT 2008. IEEE*, IEEE, pp. 181–184.
16. Liu, Z., Li, P., Zheng, Y., & Sun, M. (2009). Clustering to find exemplar terms for keyphrase extraction. *Proceedings of the 2009 Conference on Empirical Methods in Natural Language Processing: Volume 1-Volume 1*, Association for Computational Linguistics, pp. 257–266.
17. Lucci, S., Cox, J. L., & Pay, T. (2018). Another perspective on ensemble methods for automatic keyword extraction. *2018 IEEE International Conference on Big Data (Big Data)*, IEEE, pp. 5424–5426.
18. Matsuo, Y. & Ishizuka, M. (2004). Keyword extraction from a single document using word co-occurrence statistical information. *International Journal on Artificial Intelligence Tools*, Vol. 13, No. 01, pp. 157–169.
19. Mihalcea, R. & Tarau, P. (2004). TextRank: Bringing order into texts. *Association for Computational Linguistics*.
20. Naidu, R., Bharti, S. K., Babu, K. S., & Mohapatra, R. K. (2018). Text summarization with automatic keyword extraction in telugu e-newspapers. *Smart Computing and Informatics*, pp. 555–564.
21. Pay, T. (2016). Totally automated keyword extraction. *2016 IEEE International Conference on Big Data (Big Data)*, IEEE, pp. 3859–3863.
22. Pay, T. & Lucci, S. (2017). Automatic keyword extraction: An ensemble method. *2017 IEEE International Conference on Big Data (Big Data)*, IEEE, pp. 4816–4818.
23. Rose, S., Engel, D., Cramer, N., & Cowley, W. (2010). Automatic keyword extraction from individual documents. *Text Mining.*, pp. 1–20.
24. Sabzevari, M., Martínez-Muñoz, G., & Suárez, A. (2018). A two-stage ensemble method for the detection of class-label noise. *Neurocomputing*, Vol. 275, pp. 2374–2383.
25. Sundarraj, R. (2002). An optimization approach to plan for reusable software components. *European Journal of Operational Research*, Vol. 142, pp. 128–137.
26. Villmow, J., Wrzalik, M., & Krechel, D. (2018). Automatic keyphrase extraction using recurrent neural networks. *International Conference on Machine Learning and Data Mining in Pattern Recognition*, pp. 210–217.
27. Wang, R., Liu, W., & McDonald, C. (2014). Corpus-independent generic keyphrase extraction using word embedding vectors. *Software Engineering Research Conference*, volume 39, pp. 1–8.
28. Yagunova, E., Pronoza, E., & Kochetkova, N. (2018). Construction of paraphrase graphs as a means of news clusters extraction. *Computación y Sistemas*, Vol. 22, No. 4.
29. Zhang, Y., Fu, K., Sun, H., Sun, X., Zheng, X., & Wang, H. (2018). A multi-model ensemble method based on convolutional neural networks for aircraft detection in large remote sensing images. *Remote Sensing Letters*, Vol. 9, pp. 11–20.

Article received on 19/01/2019; accepted on 12/02/2019.  
Corresponding author is Tayfun Pay.



# Anaphoric Connectives and Long-Distance Discourse Relations in Czech

Lucie Poláková, Jiří Mírovský

Charles University, Faculty of Mathematics and Physics,  
Institute of Formal and Applied Linguistics, Prague,  
Czech Republic

{polakova, mirovsky}@ufal.mff.cuni.cz

**Abstract.** This paper is a linguistic as well as technical survey for the development of a shallow discourse parser for Czech. It focuses on long-distance discourse relations signalled by (mostly) anaphoric discourse connectives. Proceeding from the division of connectives on “structural” and “anaphoric” according to their (in)ability to accept distant (non-adjacent) text segments as their left-sided arguments, and taking into account results of related analyses on English data in the framework of the Penn Discourse Treebank [3, 11], we analyze a large amount of language data in Czech. We benefit from the multilayer manual annotation of various language aspects from morphology to discourse, coreference and bridging relations in the Prague Dependency Treebank 3.0. We describe the linguistic parameters of long-distance discourse relations in Czech in connection with their anchoring connective, and suggest possible ways of their detection. Our empirical research also outlines some theoretical consequences for the underlying assumptions in discourse analysis and parsing, e.g. the risk of relying too much on different (language-specific?) part-of-speech categorizations of connectives or the different perspectives in shallow and global discourse analyses (the minimality principle vs. higher text structure).

**Keywords.** Anaphoric connectives, long-distance discourse relations.

## 1 Introduction

In the area of discourse coherence research, the so-called anaphoric connectives (ACs) represent a unique phenomenon, as they combine two pillars of coherence: as discourse connectives, they connect two text units – arguments expressing

abstract objects [1] – and express a type of meaning between them (e.g. causality, conjunction, contrast, generalization), compare Example 1 with the connective *přesto* (*nevertheless*) and the meaning of concession.

- (1) *Kapacita sálu musela být rozšířena o 150 míst, tj. na 700 sedadel. **Přesto je zájem třikrát vyšší.***

*[The capacity of the hall had to be expanded by 150 seats, i. e. to 700 seats. **Nevertheless, the demand is three times higher.]**<sup>1</sup>*

At the same time, the connectives act as event anaphors, taking their left-sided argument anaphorically, which also means the possibility of long-distance discourse relations. We follow the distinction in [17] of “structural” and “anaphoric” (non-structural) discourse connectives according to their syntactic relations to either both of their arguments (subordinating and coordinating conjunctions like *because*, *although*, *and*, *but*), or to only one of them (mainly sentence adverbs, according to the prevalent classification in English, e.g. *however*, *therefore*, *instead*).

Discourse connectives<sup>2</sup> are typically located within one of the two discourse arguments they connect (the internal argument), the other argument is called external<sup>3</sup>.

<sup>1</sup>As a typographical convention for examples of discourse relations, the left-sided argument is highlighted in italics, the right-sided argument in bold and the connective is underlined.

<sup>2</sup>In this paper, we only focus on primary discourse connectives [15].

<sup>3</sup>or Arg1 according to Penn Discourse Treebank 2.0 annotation of inter-sentential relations

Arguments of structural connectives in inter-sentential relations are determined by syntactic rules and thus they are both relatively easily retrievable. Non-structural connectives provide an anaphoric link to their antecedent, i.e. the first discourse argument in the linear order, the external argument. Most often the external argument directly precedes the sentence including the AC, but non-adjacency (a long-distance discourse relation) is also possible, compare Example 2 from the Czech corpus data.

- (2) *Vedení Pojišťovny Investiční a Poštovní banky nás upozornilo, že jejich pojišťovna nebyla zařazena mezi ty, které umožňují úrazové připojištění, ač tuto službu poskytují. Omlouváme se za toto nedoplnění, dotyčná redaktorka byla pokutována. Informaci o úrazovém připojištění v Pojišťovně IPB tedy doplňujeme.*

*[The management of the insurance company notified us that their insurance company was not listed among those that allow accident insurance, although they provide this service. We apologize for this mistake, the editor in question was fined.*

**We therefore complete the information on accident insurance in the insurance company.]**

The possible non-adjacency of the external argument has been a known issue in discourse analysis and parsing (e.g. [6, 11, 5]). If a discourse parser applies the default strategy (choosing the immediately preceding sentence as the external argument) with anaphoric connectives, it may lead to incorrect results.

The aim of this paper is to study properties of ACs and long-distance relations in Czech empirically in large extent on discourse-annotated data and draw possible conclusions for automatic identification of the text units (arguments) entering discourse relations. This is a crucial task, since the correct understanding of text meaning presumes the knowledge of which parts of the text actually enter the relations.

## 2 Language Data and Tools

The dataset used in this study, the Prague Dependency Treebank 3.0 (PDT 3.0; [2]), contains

approx. 50 thousand sentences of Czech journalistic texts annotated manually on several layers of language description [4]. Annotations “beyond the sentence boundary” include discourse relations (with connectives, arguments and semantic types), pronominal and nominal coreference, bridging relations and genres of corpus documents [18]. The annotation of discourse relations was to a great extent inspired by the Penn Discourse Treebank 2.0 lexical approach (PDTB 2.0, [12]). The Prague approach [10] follows the PDTB style in marking discourse connectives as lexical anchors of local coherence relations.

The connective signals the sense of the discourse relation; if it is absent, the relation is called implicit. The list of types of discourse relations in the Prague scheme is close to the list of senses used in the PDTB (especially to the PDTB 3.0 hierarchy), slightly adopted according to the Czech syntactic tradition (there is e.g. a relation of gradation). Contrary to other approaches, the annotation was carried out directly on top of deep syntax dependency trees. Whereas discourse relations according to the PDTB can be embedded and form hierarchical structures, there is no claim about the shape of the overall structure of the text, that is why it is referred to as a framework for “shallow” discourse analysis.

For browsing, editing and searching in the data, the customizable tree editor TrEd [8] and the advanced search tool PML-Tree Query (PML-TQ; [9]) were used. The PML-TQ provides a powerful query language and as a query result offers not only individual positions in the data for a detailed inspection, but also complex statistical summaries defined by a system of output filters.

## 3 Anaphoric Connectives with a Non-Adjacent External Argument

Overall, out of the 18,072 discourse relations in the Prague Dependency Treebank 3.0<sup>4</sup> (out of which 5 455 relations are inter-sentential), 636 relations (11.7% of inter-sentential relations and

<sup>4</sup>All reported numbers correspond to 9/10 of the whole PDT 3.0 data (i.e. 44 thousand sentences), as the last 1/10 of the data has been designated as evaluation test data.

**Table 1.** 20 connectives with most occurrences in long-distance relations in the PDT 3.0, their *prevalent* translation, PoS, occurrences in long-distance relations and their proportion in inter-sentential and in all discourse relations

connective	PoS	distant	all inter	distant in inter	all	distant in all
však [however]	Coord	113	1,120	10%	1,356	8%
také [also]	Adv	54	201	27%	208	26%
ale [but]	Coord	37	376	10%	1,134	3%
dále [next]	Adv	37	104	36%	110	34%
pak [then]	Adv	31	191	16%	257	12%
tedy [so]	Coord	30	239	13%	269	11%
a [and]	Coord	27	313	9%	5,128	1%
naopak [on the contrary]	Adv	27	108	25%	134	20%
rovněž [also]	Adv	26	91	29%	97	27%
proto [therefore]	Coord	22	307	7%	339	6%
ovšem [however]	Coord	21	200	11%	257	8%
i [also]	Coord/Part	17	56	30%	73	23%
navíc [moreover]	Adv	15	145	10%	169	9%
totiž [actually]	Coord/Part	13	385	3%	405	3%
zároveň [at the same time]	Adv	12	71	17%	81	15%
přitom [and/yet]	Adv	10	156	6%	162	6%
například [for example]	Adv	8	78	10%	87	9%
zase [again]	Adv	8	32	25%	38	21%
ani [neither]	Coord	8	17	47%	35	23%
přesto [yet]	Adv/Coord?	7	79	9%	89	8%

3.5% of all discourse relations) were detected where the external argument of a connective is non-adjacent to the internal argument. Detailed figures for the most frequent connectives in long-distance relations (Table 1) show that the individual proportions range up to 47% in all inter-sentential relations.<sup>5</sup>

### 3.1 Anaphoric Connectives and PoS

Surprisingly, among the 20 most frequent Czech connectives with a non-adjacent external argument, 10 are coordinating conjunctions,<sup>6</sup> which are structural connectives and should not accept non-adjacent external arguments.

There are several possible explanations for this behaviour. First, the issue may lie in the definition of a coordinating conjunction itself

<sup>5</sup>and 36% in all relations

<sup>6</sup>3 of those 10 in fact function as connectives with two different PoS labels, according to the PDT tagging.

in different languages. There is a well-known tendency in the diachronic development of some adverbs, possibly in connection with demonstrative pronouns, towards sentence adverbs and gradually to conjunctions (see e.g. [18], p. 153–155).<sup>7</sup> In contrast to English grammar, where the strict coordinating conjunction category only contains *and*, *but* and *or* (e.g. [14], p. 920), the tradition of Czech PoS categorization also includes historically adverbial/pronominal expressions, the syntactical behaviour of which is nevertheless in contemporary Czech equal to those of conjunctions.

Second, for the task of Arg1 detection in [11], the sentence-initial *But*-adverbial was introduced, as also the annotations confirm long-distance relations for even the basic coordinating conjunctions. In the PDT 3.0, a very frequent coordinating conjunction *však* [*but*, *however*] is to our surprise

<sup>7</sup>traceable by their position in the sentence – moving from right to left, writing (separate vs. as one word), loss of original meaning etc.

more frequently used as an inter-sentential (1,120) than intra-sentential (236) connective. Moreover, in absolute numbers it is the most frequent connective with non-adjacent external argument (113 tokens) in the corpus.

Third, according to [17], structural discourse connectives allow “stretching”, similarly as syntactic dependencies within a sentence allow long-distance by embedding constituents. The interpretation may also be that structural connectives allow non-adjacent external arguments via (syntactic) stretching, not via anaphora resolution. Also another study of (German) ACs reports that the absence of an explicitly-anaphoric morpheme in the connective does not exclude its anaphoric behaviour [16].

As a practical application here, we suggest (and the more for experiments with non-English data) to also work with coordinating conjunctions as possible anaphoric connectives and to be critical to the outcomes of a PoS tagger. Also, detection of such inter-sententially used conjunctions might be not trivial, as, at least in Czech, they may not stand at the sentence-initial position, see Example 3.

- (3) “Já to nevyhrál za svých šestnáct let závodění, *já totiž žádné peníze nikdy nedostával*. Za různé prémie a etapová vítězství jsem ovšem měl tolik aktovek a necesérů, že bych je mohl prodávat. Také nějaké ty tepláky jsem vyhrál,” vzpomněl Veselý. “**Na jednu stovku si ale přece jen dobře pamatuji**.”

[“I did not win it in my sixteen years of racing, *I never got any money at all*. For various bonuses and stage victories, I have won so many briefcases and washbags that I could sell them. I’ve also won some sweatpants,” Veselý remembers. “**But those hundred crowns, I still remember them well.**”]

Lit.: On one hundred reflex.pron but still well I-remember.

### 3.2 Types of “Gaps”

For a more detailed insight, we analyzed 245 tokens of the most frequent connectives with non-adjacent discourse arguments manually (70 tokens of *však* and all tokens of *ani*, *dále*, *také*,

*ale*, *přesto*, *proto* and *přitom*), according to their relative frequencies and across semantic classes. We concentrated on their positions with respect to paragraph boundaries, reported speech zones and we classified the nature of the “gaps”, i.e. the text segments left out of the relation. Our observations are displayed in Table 2.<sup>8</sup> The detailed corpus analysis reveals that long-distance relations in the PDT 3.0 can be divided into two general groups of thematic patterns (or progressions): First, it is mostly a general statement/claim in the external argument, a certain type of *elaboration* in the gap, and a return or strong link to the first topic in the internal argument. Often, the elaboration in the gap zooms in to a specific detail or background information or gives an example.

The second group are *digressions* in the gaps. It is marked parentheses (in brackets, dashes), but much more often unmarked, and so difficult to detect, comments on the topic by the writer or other person, switching between the plan of the writer and the plan of reported content (reported speech appears in the journalistic data of the PDT often without quotation marks), and also technical digressions like author names, photo captions, subheadings.

The practical difference between these two types of gapping is their referential linkage to their closest text environment. For digressions, less coreference and associative anaphora is expected, sometimes even none (see Section 3.4 below).

### 3.3 Local Coherence and Higher Discourse Structure

It can be supposed that arguments of connectives in paragraph-initial sentences are more likely to be distant, but also to be represented by larger blocks. For the long-distant relations in the PDT 3.0, a connective in paragraph-initial (ParInIt) sentence takes another ParInIt sentence as its argument in 15.1% (96/636), and 18.4% (53/288) in the subset described in Table 2. In these specific cases it can be very difficult to decide, whether they are indeed long-distance discourse relations or whether to interpret them as relations between higher discourse segments (paragraphs) that are

<sup>8</sup>The figures for *však* contain all its 113 occurrences.

**Table 2.** Selected connectives with non-adjacent arguments: their *prevalent* semantic types, position in a paragraph-initial (PI) sentence, external arguments also in a PI sentence (PI→PI), in a paragraph-non-initial (PNI) sentence and in other settings (technical digressions, errors in annotation etc.)

Connective	Type(s)	PI (PI→PI)	PNI	Other
ani [neither]	conj	2 (2)	4	2
dále [next]	conj	19 (13)	16	2
také [also]	conj	15 (10)	35	4
však [however]	opp	39 (21)	55	19
ale [but]	opp	9 (3)	16	12
přesto [yet]	conc	3 (1)	2	2
proto [therefore]	reason	5 (2)	9	8
přitom [and/yet]	conj/opp	1 (1)	6	3

in fact adjacent. The issue in the local coherence annotation in the PDT may be the annotation rule called *the minimality principle*: annotators were instructed to include in an argument as many clauses and/or sentences as are *minimally required* and *sufficient* for the interpretation of the relation. In the PDT, no supplementary information was annotated (compare [13], p. 14), which could potentially lead to misinterpretation of cases of paragraph coherence. It is nevertheless a problem of analytical perspective, a point where local and global discourse analyses clash and each such case should be judged individually. In the studied dataset, at least 9 relations had both relevant interpretations and there may be more.

### 3.4 Non-Adjacency across Semantic Classes

The distribution of the four main semantic classes (Temporal, Expansion, Comparison, Contingency) in long-distance relations in the PDT 3.0 is very uneven. There are only 42 (6.6%) Temporal and 71 (11.2%) Contingency relations, whereas the relations of Expansion and Comparison with 261 occurrences (41%) and 262 (41.2%) are much more frequent. Additive and contrastive connectives are thus much more likely to take part in these relations, but also, from the viewpoint of a global analysis (e.g. RST, [7]), these types of

connectives can be expected more often in ParInit positions or even relating individual paragraphs. These findings correspond to the nature of the relations: causal, conditional or temporal relations require proximity of their arguments. This is often secured by syntax and by the use of subordinating conjunctions, and inter-sententially by adjacency. Although long-distance is also possible, these relations appear, at least in the studied data, less flexible to embedded contents. Furthermore, arguments of the additive connectives in our survey *dále* [next, further] and *také* [also, too] show specific patterns which relate to semantics of the relation: they have parallel syntactic patterns with referential identity of subjects (that might be interrupted in the gap) or they contain identical or synonymous verbs forms. *Dále* takes part in 14 cases of the type *He said – He further commented* and in 13 enumerative-like structures with sequences like *First – then – next*.

## 4 Conclusions

In the texts of Prague Dependency Treebank 3.0, long-distance discourse relations represent 11.7% of inter-sentential relations. In order to contribute to the automatic identification of their external arguments, we have provided a detailed linguistic analysis of connectives, arguments and semantic types in these relations and of the gaps, i.e. text segments left out of the relation. We have addressed the adverbial (anaphorical) behaviour of coordinating conjunctions, as they regularly take non-adjacent arguments (more than 290 tokens in our data).

There is also no correlation in Czech between the anaphoricity of a connective and explicitly present demonstrative morpheme in its form. Further, we have classified the gaps as either *elaborations* – giving details, examples, diverting gradually from the original topic; *digressions* – outside comments, parentheses, technicalities; or, in case of both arguments located in two paragraph-initial sentences, possibly not gaps at all. The nature of the gap can be (apart from interpunction signs) traced by different coreferential environment and thematic progressions.

Additive connectives moreover show a clear tendency to syntactic parallelism in their arguments, with referential identity of subjects, verb synonymy and high occurrence in enumerative structures. Contingency and Temporal relations (and connectives) are non-adjacent only rarely (6.6 and 11.2%). In future research, we want to focus on unmarked elaborations and comments (reported speech segments) in more detail and implement a more complex heuristics for coreference and associative anaphora in non-adjacent arguments.

## Acknowledgments

This work has been supported by projects GA17-06123S and GA19-03490S of the Czech Science Foundation. The work has been using language resources and tools distributed by the LINDAT/CLARIN project of the Ministry of Education, Youth and Sports of the Czech Republic (projects LM2015071 and OP VVV VI CZ.02.1.01/0.0/0.0/16 013/0001781).

## References

1. Asher, N. (1993). *Reference to Abstract Objects in Discourse*. Kluwer Academic Publishers, Norwell.
2. Bejček, E., Hajičová, E., Hajič, J., Jínová, P., Kettnerová, V., Kolářová, V., Mikulová, M., Mírovský, J., Nedoluzhko, A., Panevová, J., Poláková, L., Ševčíková, M., Štěpánek, J., & Zikánová, Š. (2013). Prague Dependency Treebank 3.0. Data/software. Charles University, Faculty of Mathematics and Physics, Institute of Formal and Applied Linguistics, Prague. Available from <<http://www.lindat.cz>>.
3. Creswell, C., Forbes, K., Miltsakaki, E., Prasad, R., Joshi, A., & Webber, B. (2002). The discourse anaphoric properties of connectives. *Proceedings of DAARC*, volume 4, pp. 45–50.
4. Hajič, J., Hajičová, E., Mikulová, M., & Mírovský, J. (2017). *Prague Dependency Treebank*, chapter 21. Springer Handbooks. Springer Verlag, Berlin, Germany, pp. 555–594.
5. Kolhatkar, V., Roussel, A., Dipper, S., & Zinsmeister, H. (2018). Anaphora with non-nominal antecedents in computational linguistics: A survey. *Computational Linguistics*, Vol. 44, No. 3, pp. 547–612.
6. Lee, A., Prasad, R., Joshi, A., Dinesh, N., & Webber, B. (2006). Complexity of dependencies in discourse: Are dependencies in discourse more complex than in syntax? *Proceedings of the 5th International Workshop on Treebanks and Linguistic Theories*, Prague, Czech Republic, pp. 79–90.
7. Mann, W. C. & Thompson, S. A. (1988). Rhetorical structure theory: Toward a functional theory of text organization. *Text-Interdisciplinary Journal for the Study of Discourse*, , No. 3, pp. 243–281.
8. Pajas, P. & Štěpánek, J. (2008). Recent advances in a feature-rich framework for treebank annotation. Scott, D. & Uszkoreit, H., editors, *Proceedings of the 22nd International Conference on Computational Linguistics*, The Coling 2008 Organizing Committee, Manchester, pp. 673–680.
9. Pajas, P. & Štěpánek, J. (2009). System for querying syntactically annotated corpora. Lee, G. & im Walde, S. S., editors, *Proceedings of the ACL-IJCNLP 2009 Software Demonstrations*, Association for Computational Linguistics, Suntec, pp. 33–36.
10. Poláková, L., Jínová, P., Zikánová, Š., Bedřichová, Z., Mírovský, J., Rysová, M., Zdeňková, J., Pavlíková, V., & Hajičová, E. (2012). Manual for annotation of discourse relations in Prague Dependency Treebank. Technical Report 47, Prague, Czech Republic.
11. Prasad, R., Joshi, A., & Webber, B. (2010). Exploiting scope for shallow discourse parsing. Calzolari, N., Choukri, K., Maegaard, B., Mariani, J., Odijk, J., Piperidis, S., Rosner, M., & Tapias, D., editors, *Proceedings of the Seventh International Conference on Language Resources and Evaluation (LREC'10)*, European Language Resources Association (ELRA), Valletta, Malta.
12. Prasad, R., Lee, A., Dinesh, N., Miltsakaki, E., Campion, G., Joshi, A., & Webber, B. (2008). Penn Discourse Treebank Version 2.0. Data/software. University of Pennsylvania, Linguistic Data Consortium, Philadelphia. LDC2008T05.
13. Prasad, R., Miltsakaki, E., Dinesh, N., Lee, A., Joshi, A., Robaldo, L., & Webber, B. L. (2007). The Penn Discourse Treebank 2.0 Annotation Man-

- ual. Technical report, University of Pennsylvania, Philadelphia.
14. Quirk, R., Crystal, D., & Education, P. (2004). *A comprehensive grammar of the English language*. Longman, London.
  15. Rysová, M. & Rysová, K. (2014). The centre and periphery of discourse connectives. Aroonmanakun, W., Boonkwan, P., & Supnithi, T., editors, *Proceedings of the 28th Pacific Asia Conference on Language, Information and Computing*, Department of Linguistics, Faculty of Arts, Chulalongkorn University, Department of Linguistics, Faculty of Arts, Chulalongkorn University, Bangkok, Thailand, pp. 452–459.
  16. Stede, M. & Grishina, Y. (2016). Anaphoricity in connectives: A case study on German. *Proceedings of the Workshop on Coreference Resolution Beyond OntoNotes (CORBON 2016)*, pp. 41–46.
  17. Webber, B., Stone, M., Joshi, A., & Knott, A. (2003). Anaphora and discourse structure. *Computational Linguistics*, , No. 4, pp. 545–587.
  18. Zikánová, Š., Hajičová, E., Hladká, B., Jínová, P., Mírovský, J., Nedoluzhko, A., Poláková, L., Rysová, K., Rysová, M., & Václ, J. (2015). *Discourse and Coherence. From the Sentence Structure to Relations in Text*. Studies in Computational and Theoretical Linguistics. ÚFAL, Praha, Czechia.

Article received on 25/02/2019; accepted on 04/03/2019.  
Corresponding author is Lucie Poláková.





# Complex Named Entities Extraction on the Web: Application to Social Events

Armel Fotsoh<sup>1</sup>, Annig Le Parc Lacayrelle<sup>2</sup>, Christian Sallaberry<sup>2</sup>

<sup>1</sup> RECITAL,  
France

<sup>2</sup> Univ Pau & Pays Adour / E2S UPPA,  
Laboratoire d'Informatique de l'Université de Pau et des Pays de l'Adour,  
France

armel@recital.ai, {annig.lacayrelle, christian.sallaberry}@univ-pau.fr

**Abstract.** In this paper, we focus on the extraction of social events in text from the web. We consider social events as complex Named Entities (NEs) i.e. NEs represented by a list of properties that can be simple values (text, number, etc.), "elementary" NEs and/or other complex NEs. Regarding the extraction of these complex NEs, our contribution focuses on the noisy context issue. Very few works in the state-of-the-art deal with this issue, and the few existing ones have limits in several contexts. We propose an original processing method based on supervised learning and patterns that makes it possible to focus property annotation on specific blocks of webpages. This process is generic and independent of the type of NE processed. We experimented and evaluated it with an example of complex NEs: social events. It appears that, in a noisy context, the results obtained with our approach considerably improve the standard process used in the state-of-the-art. The work was conducted with the objective of generalize it for other categories of complex NEs.

**Keywords.** Information extraction, complex named entities, social event.

## 1 Introduction

Recent developments in information technologies have made the web an important data source. Additionally, the number of contributors to this data source is increasing very rapidly and the published content is usually unstructured.

There are standards such as schema.org microdata [14] that have been defined to structure the information published on the web. However, these standards are only used very little in practice (less than 1% of websites use such standards).

Therefore, automatic processing of webpages for information extraction purposes is a difficult task. This is one reason for the rapid growth in the number of research works related to Information Extraction (IE) [4] in the textual content of webpages.

The information contained on the web generally refers to real-world objects such as people, places, points of interest (POI) or even events: in the state of the art, these objects correspond to Named Entities (NEs) [23].

However, a NE such as a social event can be described by a list of properties (e.g. its *title*, its *category*, its *location* and even its *performer*). In our approach, a NE, when described by several properties is called a *complex NE*.

Consider an excerpt of text from a webpage describing an event (see Figure 1). The extraction of different items of information on this event raises several issues. First of all, text phrases corresponding to properties of the event (*performer*, *category*, *location*, etc.) are drowned in the text without particular markups [19]. Furthermore, some properties can be expressed in

non-regular forms [8] (e.g., the event's title). Finally, some properties can be noisy.

This is the case of the event's performers, who are person (*Jeff Panacloc*, *Jean-Marc*). The text describing the event also contains other people (*Nicolas Nebot* and *Erwan Champigne*) who have nothing to do with the event performers. In the same way, the event date (*Saturday, October 14, 2017*) is described as noisy because, in the text, we have other dates as the ticketing opening date (*September 06, 2017*) and the date of another event with the same performer (*April 10, 2016*).

Ultimately, a complex NE's property is said to be noisy if the analysed text contains several candidate phrases that can match this targeted property. When this issue arises, the extraction context is referred as noisy. This paper will focus on this issue.

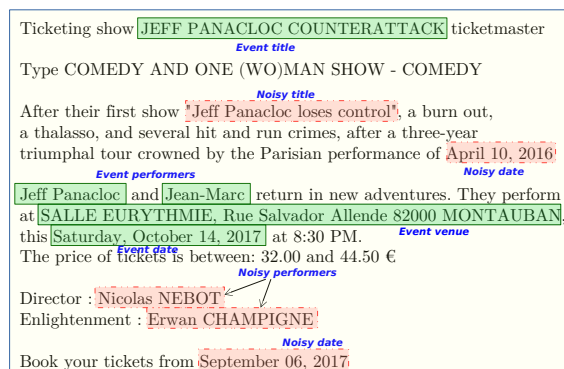


Fig. 1. Excerpt of text describing an event

We propose an approach in order to automatically extract complex NEs in webpages. The originality of this process lies in the fact that we have developed a processing method dedicated to the identification of noisy properties. This processing is based on supervised learning and patterns. It helps to isolate blocks of webpages potentially containing the properties of complex NEs. Only these blocks will then be analysed in order to identify the corresponding properties. We apply this generic approach to social events in order to evaluate the contribution of our proposal. The paper is presented as follows. We define the Named Entity concept in section 2, with a focus on

the social event NEs. We look at works carried out on complex NE extraction in section 3.

In section 4, we present in detail our extraction process for social events, while section 5 focuses on the evaluation of the obtained extraction system. Section 6 concludes the paper and proposes new research perspectives.

## 2 Definitions

The notion of Named Entity (NE) emerged during the MUC (Message Understanding Conferences) [13] in the 1990s. Despite the various changes that it has undergone since then, this notion remains difficult to define, with several definitions being proposed [13, 5, 23]. Dupont [9] defines a NE as a linguistic unit of a referential nature which refers to people, organizations, places, dates, etc. In other words, a NE can therefore be defined as a linguistic unit (phrase), uniquely identifiable in a specific context and that refers to a real-world object. To process NEs, we organize them into two main categories: *elementary* and *complex*.

Elementary NEs are represented by a single phrase that refers to the object of the real world to which it corresponds. Therefore, an elementary NE can be a person (*Gustave Eiffel*), a place (*Liberty Island*), or a date (*Saturday, October 14, 2017*), when described by one property only.

A complex NE is a NE represented by a list of properties. Consider the NE corresponding to the movie "Star Wars". It can be described by 3 properties: its title (Star Wars), its director (Georges Lucas) and a set of categories (Action, Adventure). In this example, the title of the movie is a text, the director is an elementary NE of the person type and the category is a set of texts. The director, who is a person NE, can also be seen as a list of properties (name, date of birth and place of birth). In this case, he is no longer represented as an elementary NE but as a complex one.

In formal terms, let us consider a complex NE  $e$  represented by  $n$  properties ( $n > 1$ ). We define it by:  $e = \langle p_1, p_2, \dots, p_n \rangle$ . The property  $p_i$  can be mono-valued or multi-valued (set). The value of property  $p_i$  can be an instance of a simple

type (text, number, etc.), an elementary NE, or a complex NE.

Before processing any complex NE, it is important to define models to represent them. In our application case, we deal with event NEs.

According to Zhang et al. [29], an event is "something that happens somewhere, at a given time and involves a certain number of actors". Two types of events are generally distinguished in the state of the art; these are facts and social events. Facts correspond to historical events, current topics, or even the episodes of a story. Social events refer to concerts, festivals, conferences, sport events, etc., to which a schedule and an audience are associated. For our application case, we deal with social events and propose a representation model for their description. This is given in Figure 2.

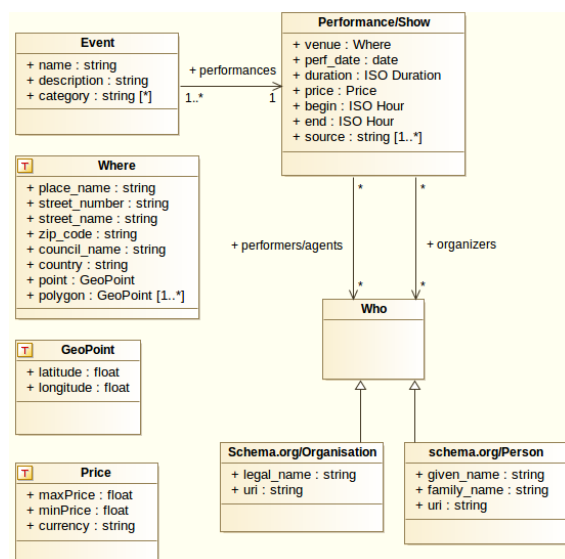


Fig. 2. Social event representation model

In our model, social events may occur on several occasions. Indeed, it is possible to attend the same concert of artist X on several dates and at different venues. For this reason, our model describes information about social events on two levels: (i) the event's thematic level, which serves to characterize it and to express its invariable properties such as the title or categories;

(ii) the event's performance, which represents variable properties.

An event's performance is scheduled and is characterized mainly by a venue, a date and a set of performers (artists). The set of performers is attached to the performance and not to the event, because the performers may vary from one occasion to another. Indeed, the actors in a play staged by a touring theatre company may change. The same applies to the organizers. Other properties such as start and end times, duration or even ticket prices are taken into account.

### 3 Related Work

Several works have focused on Complex NE extraction in text from the web. Rae et al. [22], for example, address the extraction of POIs in web content, while Orlando et al. [21] and Foley et al. [10] focus on events. The proposed extraction processes are specific to the complex NE types analysed (POIs, events, etc.). However, the extraction process is usually carried out in two steps: (i) the text is processed in order to identify the different properties; (ii) and the identified properties are gathered to build complex NEs.

Regarding the first step, four main categories of approaches are generally used: pattern-based [2, 1, 26], knowledge-based [28, 25], learning-based [17, 24, 31] and hybrid [7, 3] which combines the previous three. We have seen in the example in Figure 1 that the properties of a complex NE are generally drowned in the text.

Therefore, to identify each of them, one or a combination of the four extraction approaches is implemented according to the specificity of the targeted property. When the targeted property follows very regular forms such as dates [26], it is usually the pattern-based approach that is implemented for their extraction. Next, when the targeted property can be described in a knowledge resource, knowledge-based approaches tend to be the most used [18]. When the targeted property cannot be characterized by regular forms (as is the case of the performers in Figure 1), a learning-based approach is generally implemented for its extraction [10].

Sometimes, some of the three first approaches are combined for property identification.

For instance, to reinforce the extraction of a property using a learning-based approach, resources can be used to introduce knowledge in the learning model training step. Rae et al. [22], for example, combine data from Wikipedia to train a learning model dedicated to the detection of POIs' locations in webpages.

Once extracted, the second step consists in gathering the identified properties in order to complete complex NE fields: here arises the noisy context issue which is only covered slightly in the literature. Indeed, most of the works related to complex NE extraction deal with not noisy corpora. As a matter of facts, for event extraction, these corpora include mostly short texts from social networks posts (Twitter, Facebook, etc.). However, Orlando et al. [21] have made a proposal for the extraction, in a noisy context, of event complex NEs such as *"Larry Page founded Google Inc. in 1998 with Offices in California"* in the print media. The extraction process is also divided into two stages: identification of properties (*"Larry Page"*, *"Google Inc"*, *"1998"*, *"California"*) and their aggregation.

Regarding the second step, they consider that the context may be noisy. After identifying the properties in text, they gather them using the Cartesian product to build candidate complex NEs. A relevance score is computed for each candidate by querying a search engine (Google) and projecting the properties of candidates on the pages yielded by the search engine. Only candidates with the highest relevance scores will be targeted as valid events. This process works properly in the context of Orlando et al. because the properties are strongly related to each other. Indeed, replacing for example *"Larry Page"* by *"Mark Zuckerberg"* in a candidate event and querying Google gives results that are far removed from the query: hence the low relevance score. This makes it possible to disqualify erroneous candidate entities.

However, in some cases, this process could lead to the extraction of the wrong complex NEs. Referring to the example in Figure 1, a candidate event built by taking the ticketing opening date as the event's opening date will obtain a good score

with Orlando's approach. Most of the webpages that describe the event in Figure 1 also contain the ticketing opening date. Therefore, querying a search engine with this candidate event and the real one will lead to close scores, despite the fact that one of the two is erroneous.

## 4 Complex Named Entity Events Extraction Process

We have designed a processing chain dedicated to the extraction of social events in webpages (see figure 3). This chain takes into account the existence of noisy properties. Contrary to the standard approach which is based on two steps (property annotation and property aggregation), we introduce a new processing step that locates the relevant webpage blocks which may contain the property's phrases (block detection) before annotating them. This is interesting as the issue of noisy properties is solved before their annotation.

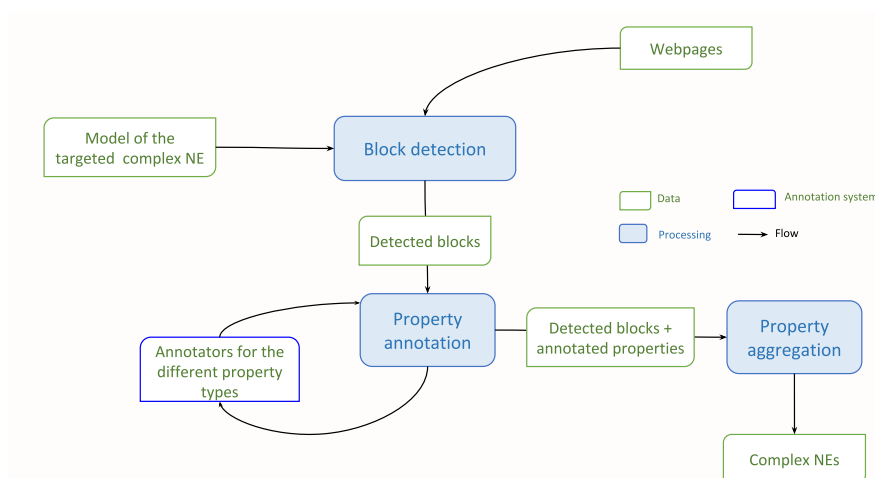
We will now detail each of the three modules of our processing chain.

### 4.1 Block Detection

The objective of this first module is to focus the annotation of properties on blocks likely to contain the right phrases. It takes as an input the social event representation model and the webpage to analyse. As an output of this first module, for each property described in the model, we have the webpage block in which it might be contained.

The observation of a sample of pages from the analysed corpus shows us that the position of the block containing properties can vary or not. For example, the event's title is always stacked in the header of the analysed webpage, for organic search reasons, while the position of category or date of events varies according to the analysed page and/or website. Therefore, we propose to use two strategies for the block detection:

- when the block containing a property is stacked in a specific position in the webpage, we propose a pattern-based approach;



**Fig. 3.** Complex NE general extraction process

- when the position of the block containing a property varies, we propose a learning-based approach to locate it.

Thus, we implement a pattern-based strategy for the event's title identification. The corresponding pattern is given below:

`< head > Title_Block < /head > .`

With regard to other properties such as category or date of events, we have carried out a supervised-learning-based strategy for their identification. We only analyse the textual content of webpages for this purpose. In the state of the art, there are several supervised learning-based algorithms that are known to be effective for text tagging. The two most commonly used ones are the Neural Networks algorithm [30] and the Conditional Random Fields (CRF) algorithm [16]. For our experimentation, we chose the CRF algorithm. This was due to the difficulty of compiling the necessary training set. In fact, Neural Networks require a large amount of data for training when the number of features to take into account is high [6]. In our context, the analysed texts are long (600 words average) and we wish to use the maximum number of features to detect blocks. CRF appears to be a good

compromise between manual annotation efforts and efficient processing.

Several criteria are necessary for determining features to infer the learning model. We based ourselves on the work of Ollagnier et al. [20] for the choice of these criteria. The most relevant for our problem are the following:

- The word (token): this will generate features like the part of speech (POS), the lemma, the shape (upper case, lower case, capitalized), the position in text, etc;
- Sequence of words: this makes it possible to take into account the co-occurrences of words;
- Properties of the words in the same window: for each word in a window, joint features are constructed from those of other words. The number of words to take into account to the left and to the right of the analysed word are parameters of the algorithm;
- Substrings of characters constituting a word: in this case, the features applied to the words are also applied to their substrings (n-gram). The maximum size of substrings to consider is a parameter of the algorithm.

We use the CRF implementation of the Stanford NER library <sup>1</sup> to train our block detection model. The training set is composed of 360 webpages from 12 ticketing websites. This corresponds to about 218,000 words as input into the trainer.

The obtained blocks are illustrated in Figure 4. Indeed, blocks containing the effective properties of the event are delimited by continuous line rectangles. In this example, the identification of the *performance block* makes it possible to distinguish between the date of the described event (*Saturday, October 14, 2017*) and both the ticketing opening date (*September 06, 2017*) and that of the other event with the same performer (*April 10, 2016*) mentioned in the text. The same block also contains information about venue, performers and even price and hour.

This block detection module needs to be tuned each time a new corpus has to be processed: pattern-based and/or learning-based approaches might be adapted.

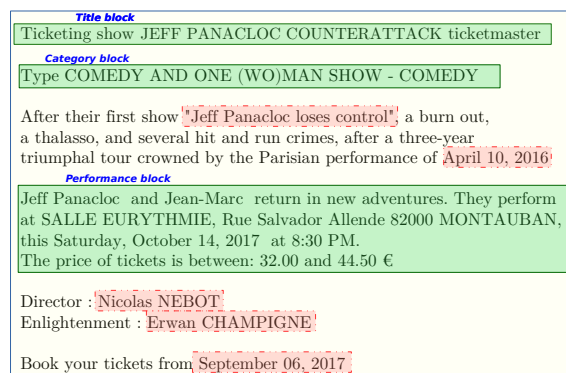


Fig. 4. Illustration of block detection

## 4.2 Property Annotation

This module is dedicated to the identification of the properties of complex NEs in the blocks detected by the first processing module. As an input, we have a set of annotators that can be state-of-the-art standards or customized ones. We also have the model describing each of the complex NE's properties and the webpage in which blocks have been detected.

<sup>1</sup><https://nlp.stanford.edu/software/CRF-NER.shtml>

For each property, the block containing it is selected first. Then an annotator in the input set is invoked according to the specificities of the property. Finally, the selected annotator is used to identify the targeted property in the relevant block.

To identify an event title in the corresponding block, we choose a supervised learning-based approach. This choice is due to the fact that this property does not follow any regular forms or cannot be expressed in a controlled vocabulary. Indeed, an event's title can be a single word (*Rihanna*) or a whole sentence (*Jeff Panacloc loses control*). We use the CRF algorithm to train our event title annotator, in the same way as for block detection. We re-use the same criteria for the generation of features, but with different parameters (window size, substring size, etc.).

The event category annotator is built using a knowledge-based approach. The main reason for this choice is that the vocabulary used to express a social event's categories is controlled. We analysed a set of online platforms dealing with this type of event to build a knowledge resource, including ticketing websites, tourist office sites and event directories. We represent this resource as a tree of concepts, each concept being characterized by a set of labels. The resulting tree has 4 hierarchical levels and contains 57 concepts for 610 labels. The resource is projected on the text of blocks to annotate the mentions of event categories.

The annotation of the event venues is based on the cascading combination of three annotators. The purpose of the cascading combination of these annotators is to detect an event venue in its most complete form whenever possible. Indeed, the first one in this combination is a pattern-based address annotator [11]. If it does not detect any place, an annotator based on a gazetteer of event venue names (built from DBPedia <sup>2</sup> and Google Maps <sup>3</sup> data) is then invoked. If no venue is detected by the first two annotators, an annotator using a city name lexicon can be used to identify the venue.

The invoked annotator for date identification is built using a pattern-based approach. This is because the dates follow very regular forms.

<sup>2</sup><http://wiki.dbpedia.org/>

<sup>3</sup><https://www.google.fr/maps>



The same is true for price and hour annotators which are also built using patterns. For the performer annotation, we use two distinct annotators.

The first one exploits a thesaurus, built from DBpedia and YAGO <sup>4</sup> data, for the identification of performer names. The second one uses a supervised learning model that is trained with the CRF algorithm. To annotate performers in relevant blocks, the first annotator is invoked, then the second is also invoked to identify performers not referenced in the resource. The combination of these two annotators is intended to minimize silence during performer annotation.

The result is shown in Figure 5.

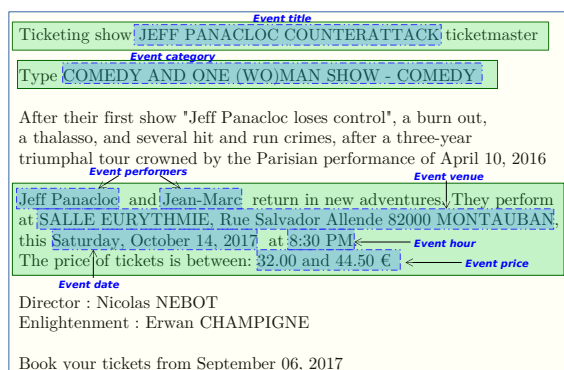


Fig. 5. Illustration of property annotation

### 4.3 Property Aggregation

Once the properties have been annotated in the corresponding blocks, the purpose of this last module is to aggregate them into the corresponding complex NEs. Two scenarios can be distinguished depending on whether the analysed webpage contains one or many complex NEs:

- If the webpage contains only one complex NE, aggregation simply associates each annotated property with the corresponding field according to the representation model of the complex NE;

<sup>4</sup><http://www.mpi-inf.mpg.de/departments/databases-and-information-systems/research/yago-naga/yago/#c10444>

- If the webpage contains many complex NEs, patterns or "dedicated" algorithms are more appropriate.

In our corpus, each analysed webpage describes a single event. Therefore, the aggregation process consists of parsing the webpage content and associating each of the annotated properties with the corresponding fields in the event's model (see Figure 2). Detecting the *performance block* makes it possible to group each of their properties (*date, venue, hour, etc.*) without ambiguity.

## 5 Evaluation

This section does not evaluate the existing complex NE annotators, nor customized ones. We simply implemented (i) the two-step state-of-the-art approach (i.e., annotation of properties first and then their aggregation) as well as (ii) our three-steps approach detailed before (Figure 3). Then, we analysed a corpus with both systems and compared the results (i.e., annotations obtained without and with a block detection module upstream). Indeed, we wanted to evaluate the ability of our annotation system to properly identify the right properties of event NEs, and more particularly to measure the contribution of the block detection module in a context where some properties can be noisy. This evaluation protocol is built based on the TREC [27] procedure.

### 5.1 Protocol

#### Scenarios

We have defined two evaluation scenarios:

- **Scenario 1:** the properties are annotated in the whole webpage text without any block detection (all the modules of the generic chain are instantiated except the block detection one);
- **Scenario 2:** the properties are annotated by analysing relevant webpage blocks. For this scenario, all the modules in our generic chain are instantiated, in particular the one dedicated to block detection.

## Evaluated Properties

Here we will evaluate the extraction of five properties: *title*, *category*, *venue*, *date* and *performer*.

## Corpus & Metrics

The evaluation corpus is composed of 150 webpages extracted from 12 ticketing websites. Each webpage describes a single event which may contain one or more performance(s). To compare our two scenarios, we use the three following evaluation metrics [12]: precision, recall and  $F_1$ -measure.

## 5.2 Results

We consider that the annotation of the properties is strongly linked to the detection of the blocks in which they are contained. As a result, we began our evaluation by measuring the quality of the block detection module. The category block and performance one (which contains the venue, the date and the performers) are characterized by the fact that their position varies in the text of the pages. Regarding the block containing the title, it corresponds to the *head* tag of the HTML page which is relatively simple and effective to detect. Our system always finds them in every page of this evaluation corpus.

Concerning the category and the performance blocks, the results of their detection on the evaluation corpus are given in Table 1. These results show that our block detection module is very accurate. So when a text block is tagged as containing a property, this is correct in more than 93% of the cases. Anyway, some blocks containing properties are not detected. This is mainly due to linguistic turns and the use of abbreviations.

**Table 1.** Evaluation of the block detection module

Blocks	Precision	Recall	$F_1$ -measure
<i>Category</i>	93.25	82.17	87.36
<i>Performance</i>	94.58	83.16	88.50

Table 2 presents the results obtained from the experimentation of the two scenarios described

above (standard approach compared with the one using the block detection module) for the property annotation.  $P$ ,  $R$  and  $F_1$  correspond to the results of scenario 1 and  $P'$ ,  $R'$  and  $F'_1$  to those of scenario 2. The gains obtained with the block detection module correspond to  $\Delta_P$ ,  $\Delta_R$ ,  $\Delta_{F_1}$ .

The first observation that emerges from these results is the clear improvement in all the  $F_1$ -measures with the instantiation of the block detection module. This is because the block detection module is working properly (about 90% correctness): when it marks a webpage block as containing a given property, this is generally correct. However, this improvement impacts the recall of certain properties, including category, venue, date and performers. Indeed, if the block containing a property has not been detected, then the associated property will not be annotated, hence the decline in the recall. This problem may arise when the block containing the targeted property is detected using the learning-based strategy (category, venue, date and performer).

Regarding the annotation of the event title, the instantiation of the block detection module serves to increase both precision and recall. Indeed, analysing only the text of the page's header considerably reduces the risk of annotating wrong titles: this explains the increase in precision. In addition, analysing just a text fragment makes it possible to take advantage of learning performance to annotate information in short texts. As a result, this increases the title annotator recall. However, some event titles are not detected by our annotator even though they are present in the header of the page. These are usually very short titles (e.g. "P. Kass"), which are very uncommon in practice.

With regard to the category, the block detection module makes it possible to improve by about 50% the precision of the system compared with scenario 1. However, the category annotator sometimes does not detect labels in the selected blocks. This is mainly because the corresponding phrases are not listed in the resource. A lexical enrichment of the category resource is underway to reduce this silence.

For the venue, the precision obtained is about 95%, well above the 83% obtained on average by Jiang et al. [15], who experimented location



Table 2. Evaluation results

Property	Scenario 1			Scenario 2			Gains		
	$P$	$R$	$F_1$	$P'$	$R'$	$F'_1$	$\Delta_P$	$\Delta_R$	$\Delta_{F_1}$
<i>Title</i>	75.26	48.67	59.11	92.57	87.63	90.03	+ 17.31	+ 38.96	+ 30.91
<i>Category</i>	42.10	92.05	57.77	92.45	76.15	83.51	+ 50.35	- 15.90	+ 25.73
<i>Venue</i>	51.02	95.28	66.46	94.25	81.16	87.22	+ 43.23	- 14.12	+ 20.76
<i>Date</i>	52.82	94.62	67.79	95.82	82.03	88.39	+ 43.00	- 12.59	+ 20.59
<i>Performer</i>	37.15	90.24	52.63	84.16	79.24	81.63	+ 47.01	- 11.00	+ 28.99

identification with several free Named Entities Recognition Systems (Stanford NER<sup>5</sup> and Spacy<sup>6</sup> in particular) on different corpora of texts. The cascade combination of three different annotators helps to limit the proportion of non-identified venues. Therefore, almost all the locations in the performance blocks are annotated by our system.

For dates, the block detection module helps to correctly isolate the relevant dates. This makes it possible to limit the proportion of erroneous dates, thus improving precision (43% gain). Moreover, as the dates follow very regular forms in our corpus, when the blocks containing them are detected, generally our annotator identifies them all.

As with the other properties, block detection helps to significantly improve the precision of performer annotation (47% gain). However, some erroneous annotations occur due to ambiguous cases. For example, suppose that the processed webpage describes a conference where "Jean Marc Verdier" is a speaker and is not referenced in the resource. Additionally, consider that the resource contains a performer whereby "Jean Marc" (the puppet of Jeff Panacloc<sup>7</sup>, a French comedian) is one of the labels: the performer annotator will tag the phrase "Jean Marc" (the puppet) as the performer, although it has absolutely nothing to do with the conference.

## 6 Conclusion

We developed an approach for the extraction of social event complex NEs in webpages.

<sup>5</sup><https://nlp.stanford.edu/ner/>

<sup>6</sup><https://spacy.io/>

<sup>7</sup>[https://fr.wikipedia.org/wiki/Jeff\\_Panacloc](https://fr.wikipedia.org/wiki/Jeff_Panacloc)

Several issues are taken into account in our proposal and the one we tackle in this paper is the noisy context problem. A complex NE's property is said to be noisy if the analysed text contains several candidate phrases that can match this targeted property. In related works, the extraction process is realized in two stages : the first one annotates the properties of complex NEs and the second one aggregates the annotated properties to build complex NEs. The noisy context issue is usually taken into account during this second stage. Our approach differs in the fact that we treat the noisy context before the annotation stage, starting by locating automatically the relevant webpage blocks which may contain the properties' phrases.

So, our processing chain consists of three main modules: (i) the first one detects webpage blocks that contain the properties of complex NEs; (ii) the second one analyses these blocks to annotate the properties; (iii) and the last module aggregates the annotated properties to build complex NEs. The introduction of the block detection module constitutes an improvement for complex NE extraction in a noisy context. Indeed, with the experimentation of our approach for the extraction of social events, we observe an average gain of 40% in precision, as well as an average gain of 25% in the  $F_1$ -measure compared to an approach without the block detection.

Our approach for social event complex NEs extraction in webpages is generic. It is important to note that the model of the targeted complex NE (e.g., social events) is one of the parameters of the block detection module (see Figure 3). However, as explained in the description of this module, each time a new corpus has to be processed,

pattern-based and/or learning-based approaches might be tuned again.

This work was conducted with the objective of replicating the process for other categories of complex NEs. To evaluate the genericity of our approach, we experimented it on another category of complex NE. A first experiment on business entities showed results similar to those obtained for social events, especially for address and activity field properties annotation. Ongoing works now focus on new experiments in order to confirm the contribution of the block detection stage. Our ultimate goal is to define a generic extraction process suited to any category of complex NE.

## References

1. Alonso, O., Baeza-yates, R., Strötgen, J., & Gertz, M. (2011). Temporal information retrieval: Challenges and opportunities. In: *1st Temporal Web Analytics Workshop at WWW*, pp. 1–8.
2. Blohm, S. (2011). *Large-scale pattern-based information extraction from the world wide web*. KIT Scientific Publishing.
3. Cellier, P., Charnois, T., Plantevit, M., Rigotti, C., Crémilleux, B., Gandrillon, O., Kléma, J., & Manguin, J.-L. (2015). Sequential pattern mining for discovering gene interactions and their contextual information from biomedical texts. *Journal of biomedical semantics*, Vol. 6, No. 1, pp. 27.
4. Chang, C.-H., Kayed, M., Girgis, M. R., & Shaalan, K. F. (2006). A survey of web information extraction systems. *IEEE Trans. on Knowl. and Data Eng.*, Vol. 18, No. 10, pp. 1411–1428.
5. Chinchor, N. & Robinson, P. (1997). Muc-7 named entity task definition. *Proceedings of the 7th Conference on Message Understanding*, volume 29.
6. Collobert, R. & Weston, J. (2008). A unified architecture for natural language processing: Deep neural networks with multitask learning. *Proceedings of the 25th International Conference on Machine Learning*, ICML '08, ACM, New York, NY, USA, pp. 160–167.
7. Dey, A., Paul, A., & Purkayastha, B. S. (2014). Named entity recognition for nepali language: A semi hybrid approach. *International Journal of Engineering and Innovative Technology (IJEIT) Volume*, Vol. 3, pp. 21–25.
8. Downey, D., Broadhead, M., & Etzioni, O. (2007). Locating complex named entities in web text. *IJCAI*, volume 7, pp. 2733–2739.
9. Dupont, Y. (2017). *La structuration dans les entités nommées*. Ph.D. thesis, Paris 3.
10. Foley, J., Bendersky, M., & Josifovski, V. (2015). Learning to extract local events from the web. *38th International ACM SIGIR Conference on Research and Development in Information Retrieval*, SIGIR '15, ACM, New York, NY, USA, pp. 423–432.
11. Fotsoh, A. T., Sallaberry, C., Le Parc Lacayrelle, A., & Moal, T. (2016). Extraction of business information on the web to supply a geolocated search service. *Proceedings of ALLDATA 2016, The Second International Conference on Big Data, Small Data, Linked Data and Open Data*, AllData '16, IARIA, Lisbon, Portugal, pp. 82–85.
12. Gleverdon, C. W. (1962). Report on the testing and analysis of an investigation into the comparative efficiency of indexing systems.
13. Grishman, R. & Sundheim, B. (1996). Message understanding conference-6: A brief history. *Coling*, volume 96, pp. 466–471.
14. Guha, R. V., Brickley, D., & Macbeth, S. (2016). Schema.org: Evolution of structured data on the web. *Commun. ACM*, Vol. 59, No. 2, pp. 44–51.
15. Jiang, R., Banchs, R. E., & Li, H. (2016). Evaluating and combining named entity recognition systems. *Proceedings of the Sixth Named Entity Workshop, Joint with 54th Association for Computational Linguistics*, pp. 21–27.
16. Klinger, R. & Tomanek, K. (2007). Classical Probabilistic Models and Conditional Random Fields. Technical Report TR07-2-013, Department of Computer Science, Dortmund University of Technology.
17. Le, N. T., Mallek, F., & Sadat, F. (2016). Uqam-ntl: Named entity recognition in twitter messages. *WNUT 2016*, pp. 197.
18. Lejeune, G., Brixtel, R., Doucet, A., & Lucas, N. (2015). Multilingual event extraction for epidemic detection. *Artificial intelligence in medicine*, Vol. 65, No. 2, pp. 131–143.
19. Nadeau, D. & Sekine, S. (2007). A survey of named entity recognition and classification. *Lingvisticae Investigationes*, Vol. 30, No. 1, pp. 3–26.
20. Ollagnier, A., Fournier, S., & Bellot, P. (2016). Cascade de crfs et svm pour la détection

- de références bibliographiques diffusés dans les articles scientifiques. *CORIA-CIFED*, pp. 139–152.
21. **Orlando, S., Pizzolon, F., & Tolomei, G. (2013).** Seed: A framework for extracting social events from press news. *22Nd International Conference on World Wide Web, WWW '13 Companion*, ACM, New York, NY, USA, pp. 1285–1294.
  22. **Rae, A., Murdock, V., Popescu, A., & Bouchard, H. (2012).** Mining the web for points of interest. *Proceedings of the 35th International ACM SIGIR Conference on Research and Development in Information Retrieval, SIGIR '12*, ACM, New York, NY, USA, pp. 711–720.
  23. **Sekine, S., Sudo, K., & Nobata, C. (2002).** Extended named entity hierarchy. *LREC*.
  24. **Shinyama, Y. & Sekine, S. (2004).** Named entity discovery using comparable news articles. *Proceedings of the 20th international conference on Computational Linguistics*, Association for Computational Linguistics, pp. 848.
  25. **Soner, K., Ozgur, A., Orkunt, S., Samet, A., K., C. N., & N., A. F. (2012).** An ontology-based retrieval system using semantic indexing. volume 37, Oxford, UK, UK, pp. 294–305.
  26. **Strötgen, J. & Gertz, M. (2010).** Heildtime: High quality rule-based extraction and normalization of temporal expressions. *Proceedings of the 5th International Workshop on Semantic Evaluation*, Association for Computational Linguistics, pp. 321–324.
  27. **Voorhees, E. M. & Harman, D. K. (2005).** *TREC: Experiment and Evaluation in Information Retrieval (Digital Libraries and Electronic Publishing)*. The MIT Press.
  28. **Wang, W. & Stewart, K. (2015).** Spatiotemporal and semantic information extraction from web news reports about natural hazards. *Computers, environment and urban systems*, Vol. 50, pp. 30–40.
  29. **Zhang, C., Zhou, G., Yuan, Q., Zhuang, H., Zheng, Y., Kaplan, L., Wang, S., & Han, J. (2016).** Geoburst: Real-time local event detection in geo-tagged tweet streams. *Proceedings of the 39th International ACM SIGIR conference on Research and Development in Information Retrieval*, ACM, pp. 513–522.
  30. **Zhang, G., Patuwo, B. E., & Hu, M. Y. (1998).** Forecasting with artificial neural networks: The state of the art. *International journal of forecasting*, Vol. 14, No. 1, pp. 35–62.
  31. **Zhou, G. & Su, J. (2002).** Named entity recognition using an hmm-based chunk tagger. *Proceedings of the 40th Annual Meeting on Association for Computational Linguistics, ACL '02*, Association for Computational Linguistics, Stroudsburg, PA, USA, pp. 473–480.

Article received on 17/01/2019; accepted on 04/03/2019.  
Corresponding author is Arnel Fotsoh.



# An Exploratory Study of the Use of Senses, Syntax and Cross-Linguistic Information for Subjectivity Detection in Spanish

Rodrigo López, Daniel Peñaloza, Francisco Beingolea, Juanjose Tenorio, Marco Sobrevilla Cabezudo

Universidade de São Paulo,  
Instituto de Ciências Matemáticas e de Computação,  
Brazil

{a20112387, daniel.penaloza, francisco.beingolea, juanjose.tenorio}@pucp.pe, msobrevillac@usp.br

**Abstract.** This work presents an exploratory study of Subjectivity Detection for Spanish. This study aims to evaluate the use of dependency relations, word senses and cross-linguistic information in Subjectivity Detection task. The first steps of this method include the labeling process of a Spanish corpus and a Word Sense Disambiguation algorithm. Then cross-linguistic English-Spanish information is obtained from Semcor corpus and used together with the Spanish data. Finally, this approach (using all gathered information and supervised algorithms) was tested showing better results than the baseline method in general.

**Keywords.** Subjectivity detection, dependency relations, wordnet, subjectivity word sense disambiguation, graphs, Spanish.

## 1 Introduction

Subjectivity Detection is the task that aims to determine whether a text is subjective or objective which means if it expresses an opinion or not [19]. According to [11], this task is considered to be more difficult than polarity classification; which focuses in recognize subjectivity as positive or negative. This could be due to different reasons such as non subjective sentences getting classified as positive or negative, an objective sentence implying an opinion which [8] describes as implicit opinion and more different cases.

Several studies on Sentiment Analysis have been performed, but most of them are in English including their tools and data [9], which is a reason to contribute with information from Spanish. Besides, classic methods attribute the subjectivity

of text to the value of its respective words; ignoring some other factors such as their respective senses or the relations between them. According to [13], sentences may have "sentiment words" but this is not enough to differentiate an opinion sentence from a non-opinion one. Considering the difficulties mentioned, some examples are shown below:

- *Mi teléfono se apaga una y otra vez.* (My cellphone turns off over and over again).
- *El nuevo Samsung Galaxy Note 7 es la bomba.* (The new Samsung Galaxy Note 7 is the bomb).
- *Había una bomba en la escuela.* (There was a bomb in school).

In the first example, an implicit opinion is shown, an objective sentence which expresses an opinion, in this case a negative one. About this sentence, it is significant to emphasize that the expression *una y otra vez* was associated with the word *apaga*, adding a new value to the sentence, making impossible to consider this sentence as an objective one. In the second example there is a subjective sentence with the word *bomba*, but the third one is an objective sentence with the same word, so a traditional classifier could have difficulties, since the senses of words are not considered.

This work presents an exploratory study of Subjectivity Detection for Spanish, which considers both the dependency relations of the words and word senses in the detection process.

Also, due to the lack of annotated resources (corpora with senses and subjectivity annotation) and in order to evaluate the cross-linguistic potential, we experimented the use of resources in English to train the subjectivity detector and compare their results with the training over a portion of an Spanish corpus manually annotated.

The paper is organized as follows, Related works are presented in Section 2. Section 3 discusses the work of gathering the knowledge for Subjectivity Detection. Section 4 is about testing the data obtained on Section 3 with supervised learning methods in order to see the subjectivity detection in sentences. Finally, Section 5 is about the final conclusions of this work.

## 2 Related Works

A semantic orientation-based approach is presented in [7]. Negation and POS-Tagging were used to choose the best features for subjectivity detection between uni-grams and phrases. SentiWordNet [1] was used with Point-wise Mutual Information (PMI) [4] to determine the semantic orientation of English documents, with good results using several features.

A study using information from Spanish tweets was proposed by [17]. Tweets include a lot of information besides the text which was exploited in this work. This information included unstructured and structured data. Thus, with these categories, different features were used such as emoticons, favorites, and retweets. After that, different supervised learning algorithms used each kind of data to get interesting results.

A framework for subjectivity detection using features in English and Spanish is shown in [3]. This framework uses Extreme Learning Machine (ELM), described in [6], but with Bayesian networks supporting its structure. Firstly, text is converted in a vector of words. This vector is processed in a deep convolutional neural network together with ELM. Finally, a Fuzzy Recurrent Neural Network is used to classify the initial text as positive, negative or neutral.

The work proposed by [14] presented an unsupervised Word Sense Disambiguation strategy for Subjectivity Detection in English. This approach

relied on labeled data from different resources and information from SentiWordNet, since its focus was to just determine their subjectivity value. After this a rule-based method to classify sentences was used counting the words and testing it with supervised classifiers.

A rule-based method which uses knowledge from WordNet [12] and SentiWordNet for texts in Spanish is presented in [2]. It includes Word Sense Disambiguation using graphs with WordNet's senses to get subjectivity values. Then each word depending on its subjectivity value and its lexical category get a different weight to determine the subjectivity of a sentence. Besides, to get some of its parameters and values this work used the Semcor corpus in order to evaluate the usefulness of resources from other languages [10].

These studies have shown some important points such as: firstly, since there are not enough information from Spanish, using English resources should help with no problems, also the use of WordNet and SentiWordNet is really common. Secondly, there are different techniques but most of them rely on just the words, which means this work will be a good contribution. Thirdly, some approaches proved that using different features besides just the text may show great results. Finally, the use of Word Sense Disambiguation is helpful and improve the results for this task.

## 3 Subjectivity Detection

This work is based on graphs and dependency relations, which are used for a Word Sense Disambiguation method. Then, these results together with the same relations are used as features for supervised learning algorithms to determine the subjectivity of the sentences. The steps required for this, are explained in the next subsections.

### 3.1 Preliminaries

#### 3.1.1 Corpus Annotation

In order to explore the subjectivity detection strategies for Spanish, the FilmAffinity corpus [2] was used. This corpus contains 2.500 objective

sentences and 2.500 subjective sentences. In this paper, we will use the subjective sentence presented in Example 1 to explain the steps performed in our work.

**Example 1.** *Este inspirador drama, mientras que trafica con clichés, logra no entregar su mensaje de una manera demasiado pesada.*

Since this corpus only contains information about subjectivity for each sentence, and our work focused on exploring the use of fine-grained information in subjectivity detection, it was necessary to incorporate more knowledge into the corpus manually. In this case, information about senses were incorporated. Senses used were extracted from Multilingual Central Repository 3.0 (MCR) [5], which includes WordNets from different European languages (including Spanish) and is aligned with Princeton WordNet 3.0 [12] and SentiWordNet 3.0 [1] (which includes polarity information to senses).

The annotation process on content words, i. e., Nouns (N), Verbs (V), Adjectives (A) and Adverbs (R) used the MCR's senses<sup>1</sup>. To determine the words belonging to these grammatical categories and its respective lemmas, Freeling 4.0 [15] was used.

Due to annotation being a long and difficult task to be performed, just a small percentage of sentences of the corpus was annotated. This represented 8% (200 objective and 200 subjective sentences) of the corpus. The annotation was performed by four annotators with knowledge about Natural Language Processing. Besides the sense annotation, information from SUMO<sup>2</sup> was extracted, taking advantages from alignments between SUMO and MCR. SUMO contains information about some kind of attributes, specifically, any word may have more multiple attributes but in this case just *SubjectiveAssessmentAttribute* was extracted for the annotation.

Some annotated tokens of the Example 1 and its respective information are presented in Table 1. Column "Sense Identifier" contains the InterLingual Index. This is the WordNet's identifier

and is useful to map between WordNets and SentiWordNet. In table 1, several senses may be seen for each word. For example, the word "drama" presents three senses associated with the synonyms "dramaturgia and dramática", "obra teatral", and "evento dramático" and "tragedia", respectively. Also, glosses and attributes from SUMO are presented for each word sense. With this information, annotators had to choose the sense more adequate in each sentence.

**Table 1.** Words and Senses Information

Word / Lemma	Tag	Sense Identifier	Synonyms	Gloss	Attributes
drama	N	spa-30-06376154-n	dramaturgia	-	Text
		spa-30-07007945-n	dramática	-	Text
		spa-30-07290278-n	obra teatral	-	SubjectiveAssessmentAttribute
demasiado	R	spa-30-00047392-r	evento dramático	-	SubjectiveAssessmentAttribute
		spa-30-00047392-r	tragedia	-	SubjectiveAssessmentAttribute
		spa-30-00415963-r	excesivamente	más de lo necesario	SubjectiveAssessmentAttribute

### 3.1.2 Subjectivity Annotation

Even though senses were annotated, this study focused on subjectivity information, therefore, information from SentiWordNet was incorporated taking advantage of the alignments with MCR. SentiWordNet is focused on sentiment analysis and assigns positive, negative and neutral scores for each word sense, which must sum to 1. Thus, subjectivity score was defined as the sum of positive and negative scores and objectivity score was defined by the neutral score. After this, four subjectivity categories (non-subjectivity or NS, low subjectivity or LS, middle subjectivity or MS, and high subjectivity or HS) were defined according the subjectivity score. Table 2 presented the range of each category. Also, senses with *SubjectiveAssessmentAttribute* were annotated as HS.

**Table 2.** Subjectivity Categories

Category	NS	LS	MS	HS
Subjectivity Score Range	0	≤ 0.25	≤ 0.50	> 0.50

### 3.2 Subjectivity Word Sense Disambiguation (SWSD)

The first step in our proposal consisted in determining the subjectivity category for each word

<sup>1</sup>Available in <http://adimen.si.ehu.es/cgi-bin/wei/public/wei.consult.perl>

<sup>2</sup>Available in <http://www.adampease.org/OP/>

in a target sentence in order to use them to determine the subjectivity of a overall sentence. To achieve this step, an adaption to the work proposed by [18] was performed because our work was focused on disambiguating senses instead words.

### 3.2.1 Graph Building

Similar to [18], the graph for a sentence were built from its dependency tree. Thus, the nodes were defined by the senses of the words (obtained from MCR and SentiWordNet) and the edges were defined by the dependency relations included in the dependency tree.

Figure 1 shows the dependency tree of the Example 1 generated by Freeling, which will be used in this Section.

In order to evaluate the level of granularity of the nodes (in relation to senses and subjectivity), two configurations for the graphs were tested. The first one, called Separated Graph, considered a word-sense for each node. The second one, called Grouped Graph, considered a group of senses with the same subjectivity category for each node.

The weight of the edges was defined as the inverse of the distance between two nodes in the WordNet knowledge graph since it was considered that two senses are more related when they are closer in a graph. Distances were obtained from the application of Dijkstra algorithm on whole WordNet Knowledge Graph.

The weights in Separated Graphs were easier to be calculated (the definition before mentioned was used), since each node contained only one sense. In the case of grouped graphs, the weights were defined as the maximum value from the relations between the senses involved in each edge.

Figure 2 shows a subgraph of the graph generated for Example 1 considering the Grouped Graph configuration. As it may be seen, nodes contains one or more senses of a word according to the subjectivity category. For example, the node of the word "logra" belonging to the Low Subjectivity category (LS) groups four senses (sense identifiers are shown in Figure 2). Also, nodes are connected to other nodes according to their dependency relation. For example, the connection between "logra" and "drama" is defined

by the dependency relation "subj" (it may be seen in Figure 1).

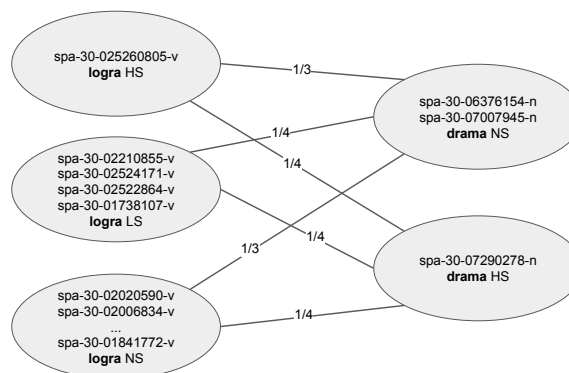


Fig. 2. Subgraph of the example sentence

### 3.2.2 SWSD

After Graph building, the subjectivity word sense disambiguation method was applied. Similar to [18], the PageRank algorithm [16] was executed. The Equation (1) shows the PageRank algorithm:

$$Pr = cMPr + (1 - c)v. \quad (1)$$

This equation is used in a graph (G) with N vertices, with these variables: The variable "Pr" will contain the result value for each vertex of the graph. The variable "c" is a constant from PageRank called damping factor. The variable "M" is an square matrix ( $N \times N$ ) with each element represented by the value  $M_{ji} = 1/d_i$  if there is a relation between vertices "i" and "j", otherwise the value is 0. The value of  $d_i$  represents the number of edges going out from the i vertex. The variable "v" is a vector containing a value for each vertex of the graph and its value is usually  $1/N$ .

In this case, an adaptation of the algorithm used in [18] was used on both graph configurations. Thus, some changes were implemented. For example, the definition of cell values of matrix "M" was changed as shown in Equation (2) :

$$M_{ji} = \frac{w_{ij}}{\sum_z w_{iz}}. \quad (2)$$



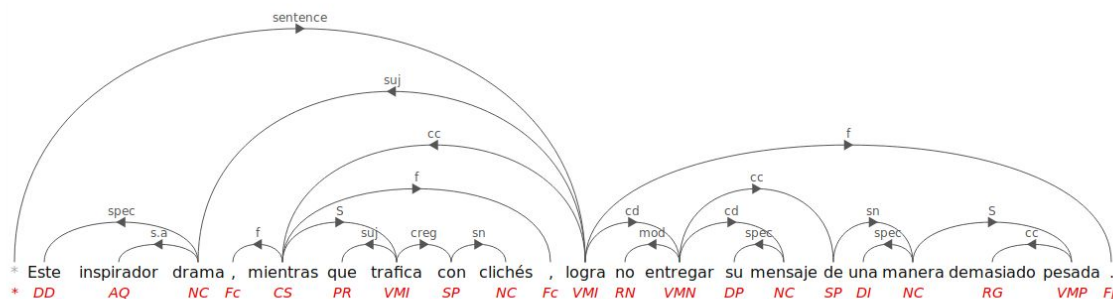


Fig. 1. Dependency Tree of Example 1

In this equation,  $w_{ij}$  is defined as the weight of the edge between vertices "i" and "j" and the sum considers all weights of edges that start from vertex "i". Besides that, the "Pr" vector was initialized with the value  $1/N$  for each vertex and vector "v" had the value of frequency obtained from the WordNet as probabilities for each vertex. Finally, the damping factor used was 0.85 and the number of iterations was 30.

### 3.3 Bringing Cross-linguistic Knowledge

Since the already described corpus annotation is an important but laborious task, gather more data was necessary. Considering there were not many resources for a non-English language, cross-linguistic knowledge was used. Besides, in Section 2 was shown that using English resources could be useful and it was an opportunity to evaluate their influence in the results.

In this case, the Semcor corpus was used. Semcor contains 20.138 sentences annotated with WordNet's senses but it is not tagged with subjectivity classification at sentence-level. This way, OpinionFinder 2.0<sup>3</sup> [20] was executed to label which sentences were objective and subjective automatically. OpinionFinder<sup>4</sup> is a sentiment analysis tool, which shows a precision (91.7%) in the Subjectivity Detection task. After the execution of OpinionFinder, 934 objective sentences and 934 subjective sentences were selected to compose the English corpus.

<sup>3</sup>Available in [http://mpqa.cs.pitt.edu/opinionfinder/opinionfinder\\_2/](http://mpqa.cs.pitt.edu/opinionfinder/opinionfinder_2/)

<sup>4</sup>This tool was used because shows good results in the Subjectivity Detection task.

To conclude with this section, it is important to note that since the tools used for this corpus were different, there were some information that were not available like it was for FilmAffinity's corpus. For example, SUMO ontology was not available neither the relations between senses from WordNet, due to labels differences; only the senses and its subjectivity values were found. Also, since there was a lot of sentences in the Semcor to check subjectivity manually, the OpinionFinder was used. So, these differences between how both corpus were worked may lead to different results, that will be describe later.

### 3.4 Incorporating Syntactic Knowledge

In order to evaluate the contribution of syntactic knowledge, several experiments were performed. Firstly, the words and their subjectivity were considered as features, since the majority of works rely on this. So, 16 features were considered, due to 4 subjectivity categories and 4 grammatical categories, being called grammatical features.

Secondly, with the obtained relations, the proposal of this work was to use them together with the words and categories as features. Mixing together the previous 16 grammatical features with each other according to their relations, resulting in 136 features called dependency features. Next, it was decided to mix both the the grammatical and dependency features to evaluate the their use. Finally, all these features were used with different supervised learning methods. Some examples of the dependency features, using Example 1, are shown in Table 3.

**Table 3.** Final Features

Relations	Features
inspirador - drama	A-HS N-NS
drama - logra	N-NS V-LS
demasiado - pesada	R-HS V-NS

## 4 Results and Discussion

### 4.1 Corpus Annotation

In relation to the corpus annotation, it is important to regard the following details:

- We used the senses from WordNet with information in Spanish, however sometimes that was not enough to choose the appropriate sense, so information from WordNet in English was checked too and when there was not enough information to resolve the confusion with similar senses, the most frequent one was consider the right one.
- There were cases when an appropriate sense could not be found for a word in both Spanish and English; so, a lemma that could be considered as a similar one, in the specific context, was used to search the right sense.
- There were words that were classified wrongly by the POS-tagger, so in those cases, the POS-tag was changed for an adequate one and the search used the lemma with its new tag, with the first and/or second consideration if necessary.
- When the first, second and third items were not enough for a word, it was left blank, since there were cases when a word has no meaning by itself (such as proper names or modal verbs) making it impossible to find any sense at all.

Finally, 4.620 words (belonging to 400 sentences extracted of the original corpus) were annotated and used to evaluate the recall of the subjectivity word sense disambiguation method (SWSD).

### 4.2 Subjectivity Word Sense Disambiguation

In relation to the SWSD, the results obtained from the SWSD method (graphs grouped and separated) were compared with a baseline. In our case, The Most Frequent Sense (MFS) was selected as baseline. This heuristic works in the following manner: All words were labeled with its most frequent sense from the WordNet. The results are shown in Table 4.

**Table 4.** SWSD summary

	Graphs					MFS		
	Group	R	Sep	R	Total		R	Total
Noun	1210	0.82	1199	0.81	1473	1817	0.83	2198
Verb	686	0.63	716	0.66	1084	750	0.65	1162
Adj.	625	0.74	623	0.74	840	665	0.74	897
Adv.	266	0.84	269	0.85	316	303	0.83	363
	2787	0.75	2807	0.76	3713	3535	0.77	4620

Table 4 shows the results of all methods tested. As it may be seen, using Separated Graphs produced better results than Grouped Graphs, even though this difference could be not significant. Also, the results of SWSD using separated graphs outperformed the results for MFS in all grammatical categories, except for nouns, producing a worse, although not significant, overall performance (due the frequency of annotated nouns).

These results may be explained by different reasons. For example, there were problems with the tools used, as explained earlier in this section, and a small amount of data could be annotated. Besides, our method could not analyze all the data, since we use relations between words, but that was not the case for MFS.

Finally, it is important to mention that the most common mistakes in the algorithms used for WSD in this work happened with verbs a significant number of times. This could be explained by all the problems already mentioned in the annotation process; such as problem with the tools (POS-tagger) or finding the appropriate sense for a word, since some words, specially verbs, are associated with a big number of senses making it more difficult for the algorithm.

### 4.3 Subjectivity Detection

As mentioned in Section 3.3 and Section 3.4, we evaluated the use of Semcor (English corpus) and the use of dependency relations in subjectivity detection task. Also, we tested the usefulness of subjectivity word sense disambiguation in subjectivity detection. Thus, we experimented the following configurations: grouped graphs and separated graphs; training on Semcor, FilmAffinity and both corpus together; and training using dependency features, grammatical features and both features together.

All experiments were performed using Linear SVM algorithm (C value was 0.01) with all features normalized (without feature selection or dimensionality reduction). Also, a non-swds baseline was used. Specifically, this baseline do not use WSD to obtain the subjectivity from words, but this is defined by the mean score of all its respective senses. Besides, we compared our results with the proposal presented in [2]. This method used an rule-based method and a subjectivity word sense disambiguation algorithm to perform subjectivity detection in the same corpus. In order to evaluate our experiments, we tested on a sub-corpus of the FilmAffinity corpus. This corpus was composed by 500 sentences (250 objective and 250 subjective). Besides, to evaluate the use of general features, another method was compared with our proposal, which used Bag of Words (BOW) and TF-IDF together with the FilmAffinity corpus.

Table 5 shows the results of all experiments performed. We may say that using SWSD and our methods specially grouped graphs show a slightly better performance than the baseline in all the experiments, except for the FilmAffinity with grammatical features which showed the best results. This may be related with nouns which have more presence in the corpus and most of them tend to be N-NS or N-HS with objective and subjective sentences respectively, according to the labeled corpus. Besides, noun senses may be from any subjectivity category, so the graph methods may be making more mistakes than taking the mean score of the senses. Then, comparing to the other works (BOW and [2]), all

our best methods (training of FilmAffinity and using grammatical features) outperformed its results, being BOW which got the worst results. One point to highlight is that work proposed in [2] used Semcor as training corpus and obtained results comparable with methods which used the same features and the same corpus.

In relation to the cross-linguistic knowledge, FilmAffinity corpus showed the best and most consistent results for all the features and methods, which showed that the Semcor corpus may not be compatible with this work. Specifically, some subjective texts in FilmAffinity corpus were composed by 2 or 4 sentences together, unlike the Semcor, where it never happened. Then, since there is a relatively difference between the size of both corpus, it was evident that Semcor had a lot more senses and dependency relations. So, considering the differences described between tools, corpus data, words and/or features; it does not seem like both corpus could be used together or that good results would come out from using the English information.

Finally, dependency features were useless in all experiments, even harming the performance when mixing with grammatical features. One possible reason is that Freeling still suffers dealing with dependency relations. Thus, we could lose lot of information from a sentence, leading to worse results.

In order to perform a deep analysis, we analyzed false positives, with some points to remark. In models with Semcor corpus most of the errors were related to features with the category HS, since other categories were dominant the presence of this could be confused easily by the classifiers. Next, with the FilmAffinity corpus, the mistakes were related specifically with the most common features from 2 categories, being A-HS and N-NS this association was really common, so it was easily confused. However, this happened in specific situations like sentences with few relations including this feature or when using words and relations together, since the words have more weight due to being more increasing the probability of errors.

After this, with both corpus together the mistakes were similar due to FilmAffinity corpus being small

**Table 5.** Subjectivity Detection Results

Method	Training Corpus	Features	Objectivity			Subjectivity			Average F1
			P	R	F1	P	R	F1	
<b>Grouped Graphs</b>	Semcor	Dependency	0.76	0.64	0.70	0.58	0.71	0.64	0.67
		Grammatical	0.58	0.78	0.67	0.84	0.67	0.74	0.71
		Dependency + Grammatical	0.66	0.69	0.67	0.71	0.67	0.69	0.68
	<b>FilmAffinity</b>	Dependency	0.89	0.70	0.78	0.62	0.85	0.71	0.76
		Grammatical	0.88	0.75	0.81	0.71	0.85	0.77	<b>0.79</b>
		Dependency + Grammatical	0.88	0.72	0.79	0.66	0.84	0.74	0.77
	Semcor + FilmAffinity	Dependency	0.76	0.75	0.76	0.74	0.76	0.75	0.75
		Grammatical	0.71	0.79	0.75	0.81	0.74	0.77	0.76
		Dependency + Grammatical	0.74	0.76	0.75	0.76	0.74	0.75	0.75
Separated Graphs	Semcor	Dependency	0.75	0.64	0.69	0.59	0.70	0.64	0.67
		Grammatical	0.56	0.78	0.65	0.84	0.66	0.74	0.71
		Dependency + Grammatical	0.64	0.68	0.66	0.70	0.66	0.68	0.67
	<b>FilmAffinity</b>	Dependency	0.88	0.70	0.78	0.63	0.84	0.72	0.76
		Grammatical	0.86	0.74	0.79	0.70	0.83	0.76	<b>0.78</b>
		Dependency + Grammatical	0.85	0.71	0.78	0.66	0.81	0.73	0.76
	Semcor + FilmAffinity	Dependency	0.77	0.73	0.75	0.71	0.76	0.73	0.74
		Grammatical	0.68	0.78	0.73	0.81	0.72	0.76	0.75
		Dependency + Grammatical	0.72	0.75	0.74	0.76	0.73	0.74	0.74
Baseline	Semcor	Dependency	0.40	0.68	0.51	0.82	0.58	0.68	0.63
		Grammatical	0.28	0.90	0.42	0.97	0.57	0.72	0.67
		Dependency + Grammatical	0.36	0.74	0.48	0.88	0.58	0.70	0.64
	<b>FilmAffinity</b>	Dependency	0.71	0.69	0.70	0.68	0.70	0.69	0.69
		Grammatical	0.77	0.81	0.79	0.82	0.78	0.80	<b>0.80</b>
		Dependency + Grammatical	0.68	0.75	0.71	0.78	0.71	0.74	0.73
	Semcor + FilmAffinity	Dependency	0.53	0.72	0.61	0.79	0.63	0.70	0.67
		Grammatical	0.49	0.85	0.62	0.91	0.64	0.75	0.71
		Dependency + Grammatical	0.58	0.75	0.65	0.81	0.66	0.73	0.70
[2]	Semcor	Grammatical	0.74	0.60	0.66	0.66	0.78	0.72	0.70
BOW	FilmAffinity	TF-IDF	0.00	0.00	0.00	1.00	0.50	0.67	0.67

in comparison, but it is interesting to note that this mix of corpus improved the results from Semcor. As a final point, it is important to mention that Semcor was checked (around 400 sentences) and a lot of mistakes were found, since most of the sentences looked like objective ones, which could be due to tools used or to the kind of text from the corpus, since it is related to news. The sentences were corrected, but with a small positive change in the results, so it confirmed that the used of Semcor was not the best for this work.

## 5 Conclusions and Final Remarks

In this paper, an exploratory study about subjectivity detection for Spanish was presented. We explored the use of Word Sense Disambiguation to identify senses' subjectivity; the incorporation of

syntactic information to subjectivity detection; and the use of cross-linguistic information, specifically English, to train supervised models for Subjectivity Detection.

The SWSD was on pair with the selected baseline, so considering that the results were not exactly bad, gathering more labeled data will be important to the evaluation of this method in order to see how the results might change in all parts of this work. Then, before considering the subjectivity detection of texts, the Semcor corpus was used for the experiments.

Considering differences in tools, data, knowledge and with the final results it was determined that the information from English was not compatible with this work, or that maybe Semcor was not an appropriate corpus due to its nature or being labeled inaccurately either by its senses

or by the OpinionFinder. Finally the experiments proposed here showed good results for the subjectivity detection task for both kind of graphs, with grouped graphs being better, proving that this approach is useful and other works will benefit from it.

Finally, some future works are related to annotate more data from FilmAffinity, to see if the results may be improved; testing data from another domain in order to see if the results change; using appropriate data from English or another language, labeling the information if necessary; and finally to use some of these features with a polarity classification tool to evaluate its usefulness.

## References

1. **Baccianella, S., Esuli, A., & Sebastiani, F. (2010).** Sentiwordnet 3.0: an enhanced lexical resource for sentiment analysis and opinion mining. *LREC*, volume 10, pp. 2200–2204.
2. **Cabezudo, M. A. S., Palomino, N. L. S., & Perez, R. M. (2015).** Improving subjectivity detection for Spanish texts using subjectivity word sense disambiguation based on knowledge. *Latin American Computing Conference (CLEI)*, IEEE, pp. 1–7.
3. **Chaturvedi, I., Ragusa, E., Gastaldo, P., Zunino, R., & Cambria, E. (2018).** Bayesian network based extreme learning machine for subjectivity detection. *Journal of The Franklin Institute*, Vol. 355, No. 4, pp. 1780–1797.
4. **Church, K. W. & Hanks, P. (1990).** Word association norms, mutual information, and lexicography. *Computational linguistics*, Vol. 16, No. 1, pp. 22–29.
5. **Gonzalez-Agirre, A., Laparra, E., & Rigau, G. (2012).** Multilingual central repository version 3.0. *LREC*, pp. 2525–2529.
6. **Huang, G.-B., Zhu, Q.-Y., & Siew, C.-K. (2006).** Extreme learning machine: theory and applications. *Neurocomputing*, Vol. 70, No. 1-3, pp. 489–501.
7. **Khanna, S. & Shiwani, S. (2013).** Subjectivity detection and semantic orientation based methods for sentiment analysis. *International Journal of Scientific and Engineering Research*.
8. **Liu, B. (2012).** Sentiment analysis and opinion mining. *Synthesis lectures on human language technologies*, Vol. 5, No. 1, pp. 1–167.
9. **Lo, S. L., Cambria, E., Chiong, R., & Cornforth, D. (2017).** Multilingual sentiment analysis: from formal to informal and scarce resource languages. *Artificial Intelligence Review*, Vol. 48, No. 4, pp. 499–527.
10. **Mihalcea, R. (1998).** Semcor semantically tagged corpus. *Unpublished manuscript*.
11. **Mihalcea, R., Banea, C., & Wiebe, J. (2007).** Learning multilingual subjective language via cross-lingual projections. *Proceedings of the 45th annual meeting of the association of computational linguistics*, pp. 976–983.
12. **Miller, G. A. (1995).** WordNet: a lexical database for English. *Communications of the ACM*, Vol. 38, No. 11, pp. 39–41.
13. **Narayanan, R., Liu, B., & Choudhary, A. (2009).** Sentiment analysis of conditional sentences. *Proceedings of the 2009 Conference on Empirical Methods in Natural Language Processing: Volume 1*, Association for Computational Linguistics, pp. 180–189.
14. **Ortega, R., Fonseca, A., Gutiérrez, Y., & Montoyo, A. (2013).** Improving subjectivity detection using unsupervised subjectivity word sense disambiguation. *Procesamiento del Lenguaje Natural*, Vol. 51, pp. 179–186.
15. **Padró, L. & Stanilovsky, E. (2012).** Freeling 3.0: Towards wider multilinguality. *LREC2012*.
16. **Page, L., Brin, S., Motwani, R., & Winograd, T. (1999).** The PageRank citation ranking: Bringing order to the web. Technical report, Stanford InfoLab.
17. **Sixto, J., Almeida, A., & López-de Ipiña, D. (2016).** An approach to subjectivity detection on Twitter using the structured information. *International Conference on Computational Collective Intelligence*, Springer, pp. 121–130.
18. **Sobrevilla-Cabezudo, M. A., Oncevay-Marcos, A., & Melgar, A. (2017).** Sense dependency-rank: A word sense disambiguation method based on random walks and dependency trees. *International Conference on Computational Linguistics and Intelligent Text Processing*, Springer, pp. 185–194.
19. **Wiebe, J. & Riloff, E. (2005).** Creating subjective and objective sentence classifiers from unannotated texts. *International conference on intelligent text processing and computational linguistics*, Springer, pp. 486–497.
20. **Wilson, T., Hoffmann, P., Somasundaran, S., Kessler, J., Wiebe, J., Choi, Y., Cardie, C., Riloff, E., & Patwardhan, S. (2005).** Opinionfinder:

A system for subjectivity analysis. *Proceedings of HLT/EMNLP 2005 Interactive Demonstrations*, pp. 34–35.

*Article received on 14/02/2019; accepted on 04/03/2019.  
Corresponding author is Rodrigo López.*

# Concept Discovery through Information Extraction in Restaurant Domain

Nadeesha Pathirana<sup>1</sup>, Sandaru Seneviratne<sup>1</sup>, Rangika Samarawickrama<sup>1</sup>, Shane Wolff<sup>1</sup>,  
Charith Chitraranjan<sup>1</sup>, Uthayasanker Thayasivam<sup>1</sup>, Tharindu Ranasinghe<sup>2</sup>

<sup>1</sup> University of Moratuwa, Department of Computer Science and Engineering,  
Faculty of Engineering,  
Sri Lanka

<sup>2</sup> CodeGen International, Trace Expert City,  
Sri Lanka

{nadeesha.14, sandaru.14, rangika.14, shanewolff.14, charithc, rtuthaya,  
tharindu.10}@cse.mrt.ac.lk

**Abstract.** Concept identification is a crucial step in understanding and building a knowledge base for any particular domain. However, it is not a simple task in very large domains such as restaurants and hotel. In this paper, a novel approach of identifying a concept hierarchy and classifying unseen words into identified concepts related to restaurant domain is presented. Sorting, identifying, classifying of domain-related words manually is tedious and therefore, the proposed process is automated largely. Word embedding, hierarchical clustering, classification algorithms are effectively used to obtain concepts related to the restaurant domain. Further, this approach can also be extended to create a semi-automatic ontology on restaurant do-main.

**Keywords.** Word embedding, word2Vec, gloVe, hierarchical clustering.

## 1 Introduction

At present, there exists an astounding amount of data available, which can be used to understand a specific domain along with the important building blocks of the domain. This data can further be used in the construction of a comprehensive ontology, which can be interpreted by the relevant and important concepts of the domain. Considering the restaurant domain, to recognize, to understand, and to evaluate the concepts and relationships among them, a proper idea on how different aspects such as food, staff, and atmosphere are

spread across the domain is required. When the aspect food is taken into consideration, there are multiple sub categories such as, seafood, meat, desserts etc.

These subcategories can be further divided forming hierarchies. Hence, the manual process of concept identification for the restaurant domain is problematic. Therefore, to avoid the hassle caused by the manual process of concept identification, research efforts have focused on semi-automatic process of concept identification for the restaurant domain.

Semi-automatic process of concept identification in a particular domain consists mainly of developing a computational model, which enables capturing the meaning of the words along with the relative meaning and similarity among words in a text corpus to identify the aspects specific to the domain and concepts which the aspects belong to. This requirement can be modeled through a high dimensional vector space, which provides a vector representation for each word in the corpus.

This numerical vector representation can be used in computing the similarity measure between the pairs of words through which, the clustering of words into similar groups can be done.

These clusters of similar words can be identified as different classes in the domain, which represent important aspects specific to that domain.

Thus, the concepts of the domain can be obtained. Further, the concepts can be ordered hierarchically so the procedure can be extended to construction of an ontology as required.

This study proposes a novel way of obtaining the concepts and aspects in the restaurant domain, semi-automatically through which, the ontology can be built. In the proposed methodology, hierarchical clustering was performed for aspects in the restaurant domain using word embedding trained through Word2Vec model. Various approaches can be adopted in order to obtain the clusters from a hierarchical clustering. This research proposes a novel way of obtaining the optimal number of clusters from a hierarchical clustering using the silhouette index. The obtained clusters are then, refined manually to identify the concepts associated with the restaurant domain through which the restaurant related entity classes are identified and these classes are available to use in the construction of the restaurant ontology.

The structure of the paper is as follows. The related work to this study is reviewed in section 2. Section 3 presents the methodology that we have adopted to develop the proposed process. Section 4 reports on the results obtained. Section 5 concludes the paper and suggests directions for future work.

## 2 Background and Related Work

### 2.1 Word Vector Embedding

First proposed by Tomas Mikolov, word vector embedding in Natural Language Processing (NLP) and feature learning is an approach of how words from a vocabulary are mapped to vectors considering the semantic meaning of words and relationship among words. This vector representation initially had a vector space with one dimension for each word. However, due to the difficulties encountered with the number of dimensions, the vector space evolved to a lower dimension.

There is a number of words embedding models like Word2vec [1], GloVe [2], La-tent semantic analysis (LSA) and Latent Dirichlet Allocation (LDA) which have their own advantages and

drawbacks depending on the task the model is being used.

LSA and LDA are popular models for statistical information related tasks while the use of these models on analogy tasks is poor [2].

Unlike LSA and LDA, Word2vec is considered to perform well in analogy tasks [2], due to its nature of considering the surrounding words and target words when creating the vectors for words. This model has two main models as Skip Gram [3] and Continuous Bag of Words (CBOW) [4]. In Skip Gram model, the context words are predicted using the target word whereas in CBOW model, the target word is predicted using the context words. GloVe model is also considered good in analogy tasks which uses a matrix factorization mechanism, combining the advantages of the Skip Gram model in Word2Vec [2] in predicting the vectors of words in text corpus. Hence, in this study, Word2Vec and GloVe models are used.

### 2.2 Clustering

Clustering is a popular field of study in data mining [5], which is used abundantly in statistical data analysis. Clustering can be defined as the mechanism of grouping a set of items, objects into clusters based on the characteristics in such a way that high intra class similarity and low inter class similarity features are preserved.

There are different algorithms, which can be used in clustering data items. But depending on the task at hand, the algorithm used can vary. For this study, hierarchical clustering is used since the number of clusters/classes that are required for identifying the concepts is not defined beforehand. There are two hierarchical clustering algorithms; agglomerative (bottom up) [6] and divisive [7] (top down). In agglomerative approach, incrementally, two clusters which are most similar are merged and hierarchical clustering is performed which is represented through a tree like structure [8] named dendrogram. In divisive approach, clusters are recursively split creating the dendrogram. In this study, agglomerative approach is used to create the hierarchical clustering because it is less time consuming and efficient to merge clusters than divisive approach [9].

Given a dendrogram, obtaining the optimal number of clusters is a challenging task for we



need to make sure the cluster quality is preserved with high intra class similarity and low inter class similarity. There are many indices that have been introduced by researchers to obtain the quality of clusters through which the number of clusters to be obtained from a dendrogram can be identified. This study incorporates the average silhouette index [10] to measure the cluster quality through which the optimal number of clusters in the hierarchical clustering can be obtained. In a previous research [11], Compact Separate Proportion (CSP) has been used for the same task.

### 2.3 Ontology Construction

Ontology can be identified as a formal representation of the concepts and relationships among the concepts in a specific domain. Manual construction of an ontology requires extensive efforts and expertise in a specific domain and is considered as an inherently complex task in the field of Natural Language and Processing. There are different approaches that have been adopted by researches in creating ontologies. One of the most common approaches is the conversion of database into an ontology [12]. Another approach is using rules to develop the ontology. These approaches have their own advantages and disadvantages. In this study, a method has been proposed to semi automate the process of identifying the concepts. It can be used to construct an ontology using word embedding and hierarchical clustering [13].

## 3 Methodology

This section discusses the methodology used for identifying the ontology entities. Each of the following subsections describe the steps of each process.

### 3.1 Data Collection & Data Pre-Processing

To conduct this research a significant amount of restaurant domain related data is required as the objective is to build a more accurate and a detailed domain specific knowledge base. For that, user reviews are an ideal method to capture every

aspect of restaurant domain like food, beverages, staff, environment etc.

Therefore, around 1 million user reviews have been obtained from different regions and different countries.

Apart from the user reviews set, another two large text corpora have been used to conduct this research. One is Stanford 10-million-word set which is freely available for research purposes and the other one is w2v\_gbg data set. For further applications, these three raw data sets were transformed into usable formats.

As the first step we eliminated all the numbers, emojis and other non-text data from the tokenized sentences from the data sets. Afterwards, stop words were eliminated since they carry less important meaning than keywords [14] and they are irrelevant for the restaurant domain. Next, lemmatization was done. The process of mapping inflected forms of a word into its base form or lemma is called lemmatization. Even after removing non-alphabetic characters and stop words from the data sets, having different forms of a word can affect the final result obtained by having different vectors for the words which are semantically equal. It also unnecessarily consumes space and affects the word embedding model. Therefore, the text corpora were lemmatized to obtain the lemma of the words. After that, all words were converted into lowercase in order to prevent duplicates of the same word with various cases.

### 3.2 Extracting Frequent Nouns

In the process of identification of the ontology classes for the restaurant domain, distinguishing the nouns that are related to the restaurant domain is vital. To obtain the domain related information, the nouns from the restaurant user reviews were used. First, 10 000 most frequent nouns from the review data set were obtained which can be identified as domain related keywords [15]. To make sure only the nouns are filtered out and to improve the accuracy, two different libraries were used. Next step was to filter the nouns whose cosine distances between them and candidate nouns were lesser than a threshold level. The candidate nouns were 'restaurant', 'food', and

'beverage' which can be identified as domain specific nouns.

Through that process, around 1500 nouns from the restaurant user reviews dataset which are more relevant to the restaurant domain were obtained.

### 3.3 Training Word2Vec and GloVe Models

In this stage, we trained word embedding models for the collected three text corpora. Restaurant review text corpus contains reviews around 1 million whereas Stanford dataset contains text around 10 million texts and the w2v\_gbg dataset contains 1 million texts.

Using GloVe algorithm, we built several GloVe models for the collected three text corpora. To build various GloVe models, different parameter values like dimension of the vector, window size, number of iterations, whether the context is symmetric or asymmetric etc. were changed. In Word2Vec, the parameters like dimension of the vector, window size, minimum frequency of a word, whether to use SG or CBOW architecture etc. were considered and several Word2Vec models were built [16, 17].

### 3.4 Hierarchical Clustering

To build a more comprehensive knowledge base for the restaurant domain, main ontology classes in the domain should be identified. Since constructing the hierarchy of concepts manually is difficult and can lead to missing out important concepts in the domain, hierarchical clustering can be used to identify the hierarchy and the clusters. The agglomerative hierarchical clustering [15] approach is adopted in this research.

To identify the domain specific concepts, around 1500 domain related keywords filtered from the 10000 most frequent nouns were used. When building hierarchical clustering, two parameters, linkage method and distance metric were used. Using different combinations of linkage method and distance metric for all the models that have been built from Word2Vec and GloVe, different hierarchical clustering was obtained and relevant dendrogram for each was obtained. It is a visualization of hierarchical relationships of the built hierarchical clustering. From all the

hierarchical clustering that have been built from each model, the most suitable model for the research was identified using the highest cophenetic coefficient (c) calculated as in equation 1:

$$c = \frac{\sum_{i < j} (x(i, j) - \bar{x})(t(i, j) - \bar{t})}{\sqrt{[\sum_{i < j} (x(i, j) - \bar{x})^2][\sum_{i < j} (t(i, j) - \bar{t})^2]}} \quad (1)$$

The terms in equation 1 are defined as follows:

- $x(i, j) = |X_i - X_j|$ ; the ordinary Euclidean distance between the  $i^{\text{th}}$  and  $j^{\text{th}}$  observations,
- $t(i, j)$ ; the dendrogrammatic distance between the model points  $T_i$  and  $T_j$ ,
- $\bar{x}$ ; average of  $x(i, j)$ ,
- $\bar{t}$ ; average of  $t(i, j)$ .

Cophenetic coefficient indicates how the word similarity compared to actual word similarity is preserved by the dendrogram and it is a linear correlation between the dissimilarity between each pair of observations and their corresponding cophenetic distances.

Comparing the cophenetic coefficient of each model, the most accurate model for the hierarchical clustering was identified with the following parameters:

- Algorithm : Skip-gram,
- Vector Size : 300,
- Window Size : 5,
- Minimum Count : 5.

### 3.5 Identifying Clusters

After the identification of most suitable dendrogram for the research purpose, the next step was to discover the clusters within dendrogram to identify the concepts in the restaurant domain. Even though it is fairly easy to obtain clusters simply by setting a horizontal cut off line at a suitable distance, it does not guarantee that it would be the optimal number of clusters that could be obtained from the dendrogram while preserving the cluster quality. So, a novel approach to measure the quality of clusters using intra-cluster similarity and inter cluster dissimilarity is used.

The silhouette index simply calculates the degree of cohesion of the objects within a cluster

compared to the other clusters. The comparison attempts to capture the degree of separation.

The index magnitude ranges from -1 to +1. Silhouette index can be calculated using any distance metric like Euclidean, Manhattan etc. Let  $i$  corresponds to a data point then  $a(i)$  is defined as the average distance from the data point to all other data points in the same cluster. The smallest average distance from  $i$  to all data points in any other cluster is defined as  $b(i)$ . Now the silhouette index can be formulated as mentioned in equation 2:

$$s(i) = \frac{b(i) - a(i)}{\max\{a(i), b(i)\}} \quad (2)$$

Using silhouette index, the average index value for a dendrogram was calculated by considering overall cluster quality. From all the calculated average indices for each possible number of clusters in the dendrogram, the maximum index value was picked. Thus, the optimal number of clusters which corresponds to the maximum average silhouette index was obtained.

### 3.6 Construction of Concept Hierarchy

Knowing the optimal number of clusters using silhouette index is a way of estimating how many leaf level concepts should be available for a given knowledge base. Since, the rough estimate of optimal number of clusters is available at this step, it is possible to construct the concept hierarchy for the knowledge base by suitably investigating the clusters. Given the rough clusters and the dendrogram of the words, identification of the concepts requires human intervention. It is not impossible to automate this process but the accuracy of identified concepts from the clusters may vastly deviate from what is actually expected. This is due to the fact that any given cluster can be conceptualized (the process of identifying the common characteristics of the words in the cluster and identifying a name which represent the group of words the best) in various ways in relation to the aspect we look into it.

E.g. the cluster of words {spoon, fork, knife} can be conceptualized differently as cutleries,

tableware etc. The dendrogram reveals how clusters are hierarchically merged.

Therefore, the rough clusters are manually checked against the dendrogram and the concept names are decided suitably for each cluster. In this process, some branches of the dendrogram appeared to be merged inappropriately. Such branches are excluded as and when necessary to preserve the cluster quality and the coherence of the concepts. Finalizing the leaf level concepts enabled the rest of the work which was to identify the next level concepts up in the hierarchy. While preserving the merging structure of the original dendrogram, the merged nodes were suitably conceptualized until the top root is arrived. At the end of the conceptualization, the concept hierarchy was obtained.

### 3.7 Classification

After identifying the concepts, it is required to classify any previously unseen words into the correct cluster/concept so that the classifier can be used to populate the ontology created from the concept hierarchy for totally unseen words. E.g., the cluster tableware includes all types of dishes and cutleries. When an unseen word like 'mug' is encountered, the classifier is expected to classify it under the tableware cluster. In order to classify the words as required, different classification algorithms were used. An artificial neural network classifier, k-nearest neighbour classifier and random forest classifier were tested with acquired cluster data.

## 4 Results

Three text corpora have been used to build Word2Vec and Glove models and dendrograms for each model have been obtained. When observing the models created from the Stanford-10-million-word set, we could identify some words which did not belong to the clusters they were in with respect to the restaurant domain. For an example, the word 'apple' was modelled as a word related to *mobile*, *WIFI*, *GSM* whereas in the restaurant domain, the word 'apple' belongs to the fruit category. The reason was, in the dataset, the word 'apple' was identified as a brand name rather than a fruit.

**Table 1.** Word2vec Model Results

Dataset	Model No.	Architecture	Cophenetic Coefficient
Stanford	W1	Skip Gram	0.494
Stanford	W2	CBOW	0.398
w2v_gbgenl	W3	Skip Gram	0.495
w2v_gbgenl	W4	CBOW	0.483
Restaurant reviews	W5	Skip Gram	0.542
Restaurant reviews	W6	CBOW	0.491

**Table 2.** GloVe Model Results

Dataset	Model No.	Context	Cophenetic Coefficient
Stanford	G1	Symmetric	0.300
Stanford	G2	Asymmetric	0.368
w2v_gbgenl	G3	Symmetric	0.401
w2v_gbgenl	G4	Asymmetric	0.460
Restaurant reviews	G5	Symmetric	0.439
Restaurant reviews	G6	Asymmetric	0.497

When w2v\_gbgenl dataset was used, the results improved comparatively to Stanford dataset, but we could identify words, which were clustered together as food, but not in specific clusters like, fruits, dessert, fish etc.

The reason was, the dataset contained very general information, which did not contain much information about the restaurant domain. Hence, when the model was created using this dataset, there was not enough information to model the vectors in such a way that clear clusters are formed.

The restaurant review dataset gave further improved and more accurate results than both the Stanford and w2v\_gbgenl datasets as it contains

more domain specific information and hence clear clusters in the dendrogram could be identified.

The models were created using the cosine distance metric as semantic similarity is considered in cosine distance unlike in the Euclidean distance. Moreover, when the cosine distance metric is used, the best method to generate the dendrogram is using the average linkage method [18].

Table 1 shows the specifications of the Word2Vec models created using the cosine distance metric, average linkage method, windows size = 5, min count = 5 and vector size = 300 along with the cophenetic coefficients obtained. Closer the cophenetic coefficient to 1, better the model is.

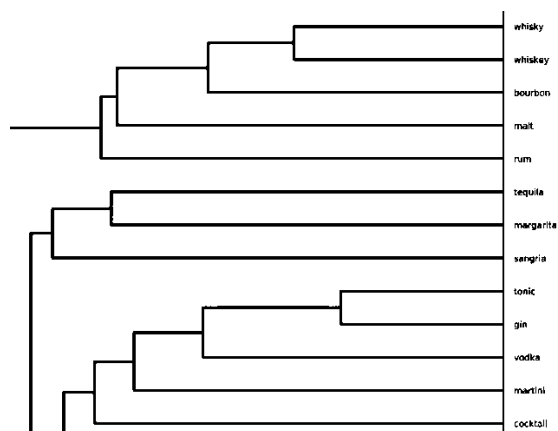
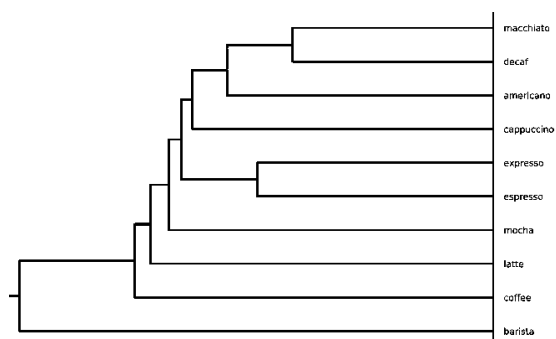
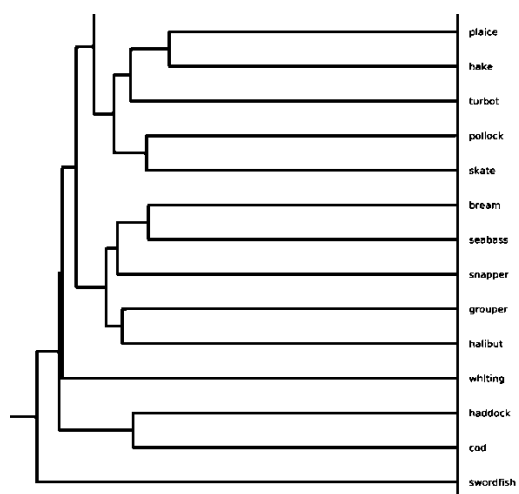
Table 2 shows the Cophenetic coefficient results for GloVe models created with the windows size = 10, min count = 5, vector size = 300 and number of iterations = 15 for models created using cosine distance metric and average linkage method.

Based on the results obtained, the most accurate model to perform hierarchical clustering was identified as W5 model. Using this model, the dendrogram for the most frequent nouns was created.

In order to obtain the optimal number of clusters, silhouette index was calculated. The maximum index value 0.35 was recorded for 96 clusters. Apparently 0.35 score is low. This is due to the fact that cluster separation is low. Even if the cluster cohesion is high, silhouette index tends to decrease when inter cluster dissimilarity drops. Since we have filtered out all the words to fit the restaurant domain, the clusters appear closely related to each other. Therefore, silhouette index has become low as expected.

In the dendrogram there are clearly separated clusters which represent various concepts related to restaurant domain. Alcoholic beverages, types of coffee, types of fish and Indian cuisine (see Fig. 1 – 4 respectively) are some identified concepts. After obtaining the dendrogram, some of the clusters were manually refined to improve the quality and the accuracy further.

Three types of classifiers have been tested with the refined cluster data obtained. Out of them artificial neural network classifier outperformed the rest. The accuracy, precision, recall and F1 scores of the classifiers are given in table 3.

**Fig. 1.** Alcoholic Beverages**Fig. 2.** Types of Coffee**Fig. 3.** Types of Fish

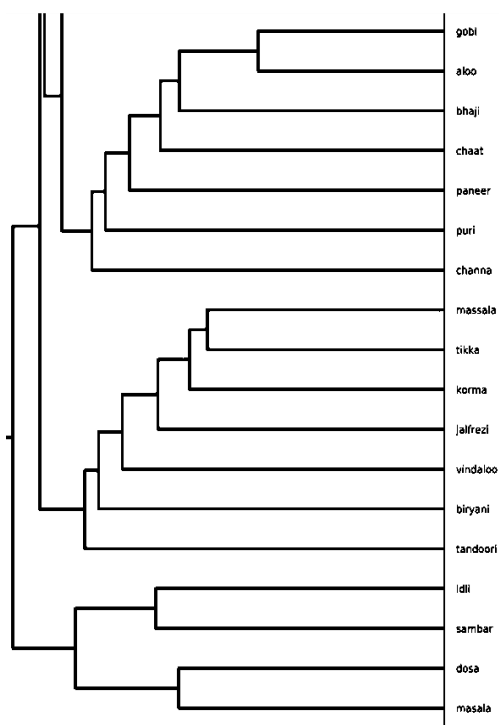


Fig. 4. Indian Cuisine

Table 3. Classification results

Classifier	Accuracy	Precision	Recall	F1 Score
Artificial Neural Network (Two hidden layers of 300 nodes each)	0.70	0.67	0.67	0.64
K Nearest Neighbor ( $k = 5$ )	0.30	0.19	0.30	0.21
Random Forest (100 estimators)	0.46	0.39	0.47	0.38

## 5 Conclusion and Future Works

This paper discusses concept identification for restaurant domain with the assistance of hierarchical clustering. In order to perform hierarchical clustering, the words should be vectorized.

Word embedding models have been used to represent words as vectors. Then, the word-vector space was clustered using hierarchical clustering. In this novel approach, the importance of automating the process of identification of concepts was outlined.

After estimating the optimal number of clusters through evaluating the average silhouette index, the classification of previously unseen words into the obtained clusters was performed.

The accuracy of the classifier is vital since it leads to the correct identification of concepts embedded in a restaurant review using the concept hierarchy obtained. The proposed methodology can also be extended to other domains without restrictions, since the process is general irrespective of the domain.

Converting the process to fit another domain is a matter of fine tuning the model to incorporate and

domain-related concepts to best suit the aspects considered. As future work, this methodology can be extended such that obtained concepts are used in ontology building through which a comprehensive knowledge base for the restaurant domain can be acquired.

## References

1. Mikolov, T., Chen, K., Corrado, G., & Dean, J. (2013). Efficient estimation of word representations in vector space, 3rd ed. *arXiv*.
2. Pennington, J., Socher, R., & Manning, C. (2014). GloVe: global vectors for word representation. *Empirical Methods in Natural Language Processing (EMNLP)*, pp 1532–1543.
3. Melamud, O., Levy, O., & Dagan, I. (2015). A simple word embedding model for lexical substitution. *NAACL-HLT*. pp. 1–7.
4. Goldberg, Y. & Levy, O. (2014). *Word2Vec explained: deriving Mikolov et al.'s negative-sampling word-embedding method*. *arXiv*.pp. 1–5.
5. Srivastava, J., Cooley, R., Deshpande, M., & Tan, P. (2000). Web usage mining: discovery and applications of usage patterns from web data. *ACM SIGKDD Explorations Newsletter*. pp. 12–23. DOI: 10.1145/846183.846188.
6. Beeferman, D. & Berger, A. (2000). Agglomerative clustering of a search engine query log. *ACM SIGKDD International Conference on Knowledge discovery and data mining*. pp. 407–416.
7. Savaresi, S., Boley, D., Bittanti, S., & Gazzaniga, G. (2002). Cluster selection in divisive clustering algorithms. *SIAM International Conference on Data Mining*. pp. 299–314. DOI: 10.1137/1.9781611972726.18.
8. Zhang, T. & Ramakrishnan, R. (1996). BIRCH: an efficient data clustering method for very large databases. *ACM SIGMOD International Conference on Management of data*. pp. 103–114. DOI: 10.1145/233269.233324.
9. Cimiano, P., Hotho, A., & Staab, S. (2004). Comparing conceptual, divisive and agglomerative clustering for learning taxonomies from text. *European Conference on Artificial Intelligence (ECAI)*. pp. 435–439.
10. Liu, Y., Li, Z., Xiong, H., Gao, X., & Wu, J. (2010). Understanding of internal clustering validation measures. *IEEE International Conference on Data Mining*. DOI: 10.1109/ICDM.2010.35.
11. Zhou, S., Xu, Z., & Liu, F. (2017). Method for Determining the Optimal Number of Clusters Based on Agglomerative Hierarchical Clustering. *IEEE Transactions on Neural Networks and Learning Systems*, Vol. 28, No. 12, pp. 3007–3017. DOI: 10.1109/tnnls.2016.2608001.
12. He, Y., Quan, T., & Hui, S. (2007). A multimodal restaurant finder for semantic web. 4th International. *Conference on Computing and Telecommunication Technologies*.
13. Recupero, D., Nuzzolese, A., Consoli, S., Presutti, V., Misael, M., Peroni, S. (2015). Extracting knowledge from text using Sheldon, a semantic holistic framework for linked ontology data. *24th International Conference on World Wide Web*, pp 235–238. DOI: 10.1145/2740908.2742842.
14. Srividhya, V. & Anitha, R. (2010). Evaluating preprocessing techniques in text categorization. *International Journal of Computer Science and Application*, pp. 49–51.
15. Beil, F., Ester, M., & Xu, X. (2002). Frequent term-based text clustering. *ACM SIGKDD international conference on Knowledge Discovery and Data Mining, ACM*, pp. 436–442.
16. Rong, X. (2014). Word2vec parameter learning explained. *arXiv*.
17. Sugathadasa, K., Ayesha, B., Silva, N., & et al. (2017). Synergistic union of word2vec and lexicon for domain specific semantic similarity. *IEEE International Conference on Industrial and Information Systems (ICIIS)*, pp. 1–6. DOI: 10.1109/ICIINFS.2017.8300343.
18. Saraçlı, S., Doğan, N., & Doğan, İ. (2013). Comparison of hierarchical cluster analysis methods by cophenetic correlation. *Journal of Inequalities and Applications*. DOI: 10.1186/1029-242x-2013-203.

Article received on 20/01/2019; accepted on 04/03/2019.  
Corresponding author is Nadeesha Pathirana.





# Deep Semantic Role Labeling for Tweets using 5W1H: *Who, What, When, Where, Why and How*

K. Chakma<sup>1</sup>, Amitava Das<sup>2</sup>, Swapan Debbarma<sup>2</sup>

<sup>1</sup> National Institute of Technology, Agartala,  
India

<sup>2</sup> Indian Institute of Information Technology, Sricity,  
India

<sup>3</sup> National Institute of Technology, Agartala,  
India

kchakma.cse@nita.ac.in, amitava.das@iiits.in, swapanxavier@gmail.com

**Abstract.** In natural language understanding, Semantic Role Labeling (SRL) is considered as one of the important tasks and widely studied by the research community. State-of-the-art lexical resources have been in existence for defining the semantic role arguments with respect to the predicates. However, such lexical resources are complex in nature which is difficult to understand. Therefore, instead of the classical semantic role arguments, we adopted the concept of 5W1H (*Who, What, When, Where, Why and How*) for SRL. The 5W1H concept is widely used in journalism and it is much simpler and easier to understand as compared to the classical SRL lexical resources. In the recent years, recurrent neural networks (RNN) based end-to-end SRL systems have gained significant attention. However, all recent works have been developed for formal texts. This paper reports on the implementation of a deep neural network using the attention mechanism for extracting the 5W1H from tweets. Our implementation reports an F-1 score of **88.21** which outperforms other recent Twitter SRL system by **28.72**.

**Keywords.** Semantic role, 5W1H, tweet, attention mechanism.

## 1 Introduction

Natural language understanding (NLU) is an important and challenging subset of natural language processing (NLP). NLU is considered as

the post-processing of text, after NLP techniques are applied on texts. Semantic Role Labeling (SRL) is a natural language understanding task that extracts semantic constituents of a sentence for answering *who* did *what* to *whom*. SRL is a shallow semantic parsing task whose primary goal is to identify the semantic roles and the relationship among them and therefore, has wide application in other NLP tasks such as Information Extraction [3], Question Answering [33, 21, 9], Machine Translation [16, 36, 38] and Multi-document Abstractive Summarization [10].

The study of semantic roles was first introduced by the Indian grammarian Panini [4] in his “*Karaka*” theory. *Karaka* theory assigns generic semantic roles to words in a natural language sentence. The relationship of the arguments with the verb is described using relations called *Karaka* relations. *Karaka* relations describe the way in which arguments participate in the action described by the verb. Several lexical resources for SRL have been developed such as PropBank [22], FrameNet [2] and VerbNet [31] that define different semantic role sets.

Gildea and Jurafsky [11] developed the first automatic semantic role labeling system based on FrameNet. Subsequent works [26, 27, 23] are considered as traditional approaches that explored

the syntactic features for capturing the overall sentence structure. Most of the SRL works are based on the PropBank [22] role set and use the CoNLL-2005 [5] shared task datasets. These datasets are mainly sections from the World Street Journal (WSJ) articles. Though there have been significant developments in studying SRL, most of the state-of-the-art SRL systems have been developed for formal texts only. However, this paper describes SRL implementation on a different genre of texts called tweets<sup>1</sup>.

Twitter is a micro-blogging site that allows a user to post texts (often known as *tweets*) within the limit of 280 characters. Tweets are often found to be informal in nature and tend to be without proper grammatical structures. Use of phonetic typing, abbreviations, word play and emoticons are very common in tweets which therefore, make it a difficult task to perform SRL on such informal texts. Let us illustrate the nature of tweets with some examples.

#### Examples:

- (1) **Abbreviation:**
  - *IMHO, Elvis is still the king of rock.*
- (2) **Wordplay:**
  - *Sometime things change from **wetoo** to #MeToo.*

In example (1), IMHO is an abbreviation for *in my humble opinion*, whereas in example (2), *wetoo* is the merger of *we* and *too*. In addition to the variations described in examples (1) and (2), users often apply creative writings such as the word *before* is often written as *b4*. These examples suggest that users are at the liberty to write tweets without following the syntactical requirements but maintaining the semantics. Following the above discussions, it appears that performing SRL on tweets is a difficult task. Therefore, the state-of-the-art SRL systems meant for formal texts, are not expected to perform well on tweets. From the available lexical resources for SRL, PropBank is the most commonly studied role set.

<sup>1</sup> Texts from [www.twitter.com](http://www.twitter.com) are known as tweets

However, annotations based on the PropBank role set requires sufficient knowledge about the constituent arguments of a predicate. Therefore, instead of using the PropBank role set, we adopted the concept of 5W1H (*Who, What, When, Where, Why, How*) as described in [6]. 5W1H concept is widely used in journalism because an article is considered complete only when all the 5W1H are present. The concept of 5W1H is similar to the *Karaka* relations and easy to understand. We discuss in detail about 5W1H in later sections.

The major contributions of this paper are:

- Development of a corpus for 5W1H extraction from tweets.
- Development of a Deep Neural Network for the 5W1H extraction from tweets.

The rest of the paper is organized as follows. Section 2 discusses the related work. Section 3 discusses the background concepts on SRL. In section 4 the deep neural network implementation is discussed. Section 5 discusses the experiments performed. Results are discussed in section 6 followed by analysis in section 7. We conclude the paper in section 8.

## 2 Related Work

Though the traditional approaches of Gildea and Jurafsky [11], Pradhan et al. [26], Punyakanok et al. [27] explored the syntactic features, recently, deep neural network based implementations have outperformed the traditional approaches. Zhou and Xu [40] were the first to build an end-to-end system for SRL, where an 8 layered LSTM model was applied which resulted in outperforming the previous state-of-the-art system. To assign semantic labels to syntactic arguments, Roth and Lapata [29] proposed a neural classifier using dependency path embeddings.

He et al. [13] developed a deep neural network with highway LSTMs and constrained decoding that improved over earlier results. To encode syntactic information at word level, Marcheggiani and Titov [19] implemented their system by combining a LSTM network with a graph

convolutional network which improved their LSTM classifier results on the CoNLL-2009 dataset.

Attention mechanism was pioneered by Bahdanau et al. [1]. Cheng et al. used [7] LSTMs and self-attention to facilitate the task of machine reading. Tan et al. [35] implemented a self-attention based neural network for SRL without explicitly modeling any syntax that outperformed the previous state-of-the-art results. Strubell et al. [32] implemented a neural network model that combines multi-head self-attention with multi-task learning across dependency parsing, part-of speech tagging, predicate detection and SRL. Their [32] method achieved the best scores on the CoNLL-2005 dataset. Liu et al. [17] are the first to study SRL on tweets. They considered only those tweets that reported news events.

They trained a tweet specific system which is based on the mapping between predicate-argument structures from news sentences and news tweets. They further extended their work in [18] where similar tweets are grouped by clustering. Then for each cluster a two-stage SRL labeling is conducted. [20] describe a system for emotion detection from tweets. Their work mainly focuses on identification of roles for *Experiencer*, *State* and *Stimulus* of an emotion. [30] proposed an SRL system for tweets using sequential minimal optimisation (SMO) [25]. Our work adopts the 5W1H extraction for SRL using deep neural network attention mechanism of Bahdanau et al. [1]. Our experiments also show that the attention mechanism is effective on the sequence labeling task of 5W1H.

### 3 Background

#### 3.1 PropBank based SRL

This section describes the SRL based on the PropBank role set. We first discuss what SRL is and then describe how the PropBank role set is applied for the SRL task. A sentence may represent an event through different surface forms. Let us consider the event of someone (John) *hitting* (event) someone (Steve).

**Example:**

(3) *Yesterday, John hit Steve with a stick*

The above sentence has different surface level forms such as:

- Steve was hit by John yesterday with a stick
- Yesterday, Steve was hit with a stick by John
- With a stick, John hit Steve yesterday
- Yesterday Steve was hit with a stick by John
- John hit Steve with a stick yesterday

In the above example, despite having different surface level representations, the event is described by the verb (*hit*) where “John” is the syntactic subject and “Steve” is the syntactic object. A subject in a sentence is the causer of the action (verb) whereas, an object is the recipient. From example (3), we are able to represent the fact that there was an event of assault, that the participants in the event are John and Steve, and that John played a specific role, the role of hitting Steve.

These shallow semantic representations are called semantic roles. For a given sentence, the objective of SRL is to first identify the predicates (verb) and the arguments; and classify those arguments of each predicate (verb). In PropBank, every verb (predicate) is described by some senses and for each verb sense, there are specific set of roles defined. For example, the verb *hit* has five different senses in the PropBank database as shown below (only two senses are shown here):

— **sense 1: hit.01, strike**

– **Role:**

- \* ARG0: hitter
- \* ARG1: thing hit
- \* ARG2: instrument hit with

– Example: John hit Steve with a stick

— **sense 2: hit.02, reach, encounter**

– **Role:**

- \* ARG0: thing hitting
- \* ARG1: thing hit

– Example: Product hit the market

Applying PropBank role set on the sentence in example (3) yields the following semantic roles:

$[Yesterday]_{\text{ARG-TMP}} [John]_{\text{ARG}_0} [hit]_{\text{V}}$   
 $[Steve]_{\text{ARG}_1} [with a stick]_{\text{ARG}_2}$

### 3.2 Modelling 5W1H

The 5W1H (*Who, What, When, Where, Why and How*) model has been attributed to Hermagoras of Temnos [28] who was an Ancient Greek rhetorician which was further conceptualized by Thomas Wilson [37]. Nowadays, 5W1H is often used in journalism to cover a report. The 5W1H are considered as the answers to a reporter's questions, which are considered as the ground for information gathering. In journalism, a report is considered complete only if it answers to the question of *Who did what, when, where, why and how*.

Let  $w = \{w_1, w_2, \dots, w_n\}$  be the sequence of words in a tweet and  $X$  be the attribute to which  $w$  is to be mapped. We therefore, assume a tweet as  $\langle w, X \rangle$ , where,  $X$  is the tuple  $\langle WHO, WHAT, WHEN, WHERE, WHY, HOW \rangle$  in 5W1H.

#### 3.2.1 Defining 5W1H

In this sub section, we define the 5W1H in line with the definitions of [39]. Let  $w = \text{"John met her privately, in the hall, on Monday to discuss their relationship"}$ :

**Definition 1: Who.** *It is the set of words that refer to a person, a group of people or an institution involved in an action.*

In  $w$ , **Who**= { John }

**Definition 2: What.** *It is the set of words that refer to the people, things or abstract concepts being affected by an action and which undergo the change of state .*

In  $w$ , **What**= { met her }

**Definition 3: When.** *It is the set of words that refer to temporal characteristics. In tweets, the notion of time may be the days, weeks, months and year of a calendar or the tick of a clock. It also refers to the observations made either before, after or during the occurrence of events such as festivals, ceremonies, elections etc.*

In  $w$ , **When**= { on Monday }

**Definition 4: Where.** *It is the set of the words that refer to locative markers in a tweet. The notion of location is not restricted to physical locations but it also refers to abstract locations.*

In  $w$ , **Where**= { in the hall }

**Definition 5: Why.** *It is the set of words that refer to the cause of an action.*

In  $w$ , **Why**= { to discuss their relationship }

**Definition 6: How.** *It is the set of words that refer to the manner in which an action is performed.*

In  $w$ , **How**= { privately }

We denote  $\psi_X(w)$  to represent the set of words contained in the text  $w$  and classified to the attribute  $X$  where,  $X \in 5W1H$ . According to the Definition 1 to 6, the 5W1H model of tweets can be represented as:

$$\psi_{5W1H}(w) = \bigcup_{X \in 5W1H} \psi_X(w). \quad (1)$$

**Table 1.** List of PropBank roles

Argument	Role
ARG0	agent
ARG1	patient
ARG2	instrument, benefactive, attribute
ARG3	starting point, benefactive, attribute
ARG4	ending point
ARGM	modifier

### 3.3 5W1H vs. PropBank

Semantic roles in PropBank are defined with respect to an individual verb sense. In PropBank, the verbs have numbered arguments labeled as: ARG0, ARG1, ARG2, and so on. In general, numbered arguments correspond to the following semantic roles shown in Table 1. Apart from the numbered arguments, PropBank also involves verb modifiers often known as the functional tags such as manner (MNR), locative (LOC), temporal (TMP) and others.

Unlike the PropBank role set, the 5W1H scheme does not specify semantic roles at fine grain levels. However, it defines a simplistic approach for extracting the key information from a given sentence (tweets in our case) for other tasks such as event detection, summerization etc. A comparison of 5W1H and PropBank is illustrated with the following examples.

#### Example:

(4) *Trump's Pyrrhic Victory Provides a BIG Silver Lining for Democrats* <https://t.co/NzO8NBBkDS>

PropBank on example (4):

— predicate: **provide**

- [Trump's Pyrrhic Victory]<sub>ARG0</sub>  
[Provides]<sub>V</sub> [a BIG Silver Lining]<sub>ARG1</sub>  
[for Democrats]<sub>ARG2</sub>.<sup>2</sup>

5W1H on example (4):

— predicate: **provide**

<sup>2</sup><https://t.co/NzO8NBBkDS>

- [Trump's Pyrrhic Victory]<sub>Who</sub>  
[Provides]<sub>V</sub>  
[a BIG Silver Lining for Democrats]<sub>What</sub><sup>3</sup>

#### Example:

(5) *One +ve I will take from Trump's victory is the acknowledged death of political correctness*

PropBank on example (5):

— predicate: **take**

- [One + ve]<sub>ARG1</sub> [I]<sub>ARG0</sub>  
[will]<sub>ARGM-MOD</sub> [take]<sub>V</sub>  
[from Trump's victory]<sub>ARG2</sub> is the  
acknowledged death of political  
correctness

— predicate: **acknowledged**

- One +ve I will take from Trump's victory  
is the [acknowledged]<sub>V</sub> [death]<sub>ARG1</sub> of  
political correctness

5W1H on example (5):

— predicate: **take**

- [One + ve]<sub>What</sub> [I]<sub>Who</sub> [will]<sub>What</sub>  
[take]<sub>V</sub> [from Trump's victory]<sub>What</sub>  
is the acknowledged death of political  
correctness

— predicate: **acknowledged**

- One +ve I will take from Trump's  
victory is the [acknowledged]<sub>V</sub>  
[death of political correctness]<sub>What</sub>

<sup>3</sup><https://t.co/NzO8NBBkDS>

**Annotation:**

Annotation based on the PropBank role set requires deep knowledge of SRL and the constituent role arguments. On the other hand, the 5W1H annotation scheme is a simplistic Q&A approach as described in our earlier work [6]. Applying the simple question of “*Who did what, when, where, why and how*” yields the constituents of the 5W1H. In example (4), for the predicate **provide**, on applying the 5W1H question of “*Who is the provider*”, yields “*Trump’s Pyrrhic Victory*” as the *Who* and the question “*What is being provided*” yields “*a BIG Silver Lining for Democrats*” as the *What*. Similarly, in example (5), we obtain the 5W1H constituents for each predicate (**take**, **acknowledge**).

However, in both these examples (4 and 5), the arguments ARG1 and ARG2 are merged as “*What*”.

Therefore, the 5W1H model does not distinguish between ARG1 and ARG2. Despite the scenario described in these examples, the constituents of the 5W1H model are mostly similar to some of the PropBank role set. A comparison between the PropBank role set and the 5W1H on our dataset is shown in Table 2. From Table 2, we observe that “*Who*” is mostly similar to ARG0 (84.48%) with a small fraction (10.34%) being similar to ARG1. This is explained with the following example.

**Example:**

(6) *Murphy Brown Comes Forward With Her Own #MeToo Story* <https://t.co/kKw81IWz5t> via @thedailybeast

In example (6), for the predicate **comes**, “*Murphy Brown* is ARG1 as per PropBank. However, if 5W1H model is applied then “*Murphy Brown* is identified as “*Who*”. An important observation is the coverage of “*What*” with ARG2, ARG4, ARGM-ADV and ARGM-MOD. The 5W1H model does not specify fine grain semantic roles as compared to the PropBank role set. This is already illustrated in examples (4) and (5). From Table 2, we also observe that “*When*”, “*Where*”, “*Why*” and “*How*” are closely similar to ARGM-TMP, ARGM-LOC, ARGM-CAU and ARGM-MNR respectively.

## 4 Deep Neural Network for 5W1H Extraction

### 4.1 Attention Background

In this section, we discuss the fundamentals of attention mechanism as proposed by Bahdanau et al. [1]. It is understandable from the example in section 3.1 that SRL is a sequence labeling task. Transforming an input sequence (source) to a new output sequence (target) is the objective of the sequence to sequence (**seq2seq**) model [34]. In **seq2seq**, both sequences can be of arbitrary lengths. The **seq2seq** model is basically an *encoder-decoder* architecture. An *encoder* encodes an input sequence and compresses the information into a context vector of a fixed length. A *decoder* is initialized with the context vector to produce the transformed output.

Under such an architecture, the decoder initial state is obtained only from the last state of the encoder network. However, there is one major disadvantage of this scheme. The fixed-length context vector is incapable of remembering long sentences. In SRL, an argument may span a long sequence in a sentence. In such a scenario, the **seq2seq** model is not suitable because it often forgets earlier parts once it completes processing the whole input. The attention mechanism was introduced by Bahdanau et al. [1] to resolve this problem. Attention mechanism creates connections between the context vector and the entire source input rather than building a single context vector out of the encoder’s last hidden state. The weights of these shortcut connections are customizable for each output element.

A sequence of dense word vectors are used to represent the input sentence. These word vectors are fed to a bi-directional long short-term memory (Bi-LSTM) [14] encoder to produce a series of hidden states that represent the input. Let us consider a source sequence  $x$  of length  $T_x$  and then use it to output a target sequence  $y$  of length  $T_y$ :

$$\mathbf{x} = [x_1, x_2, \dots, T_x], \mathbf{y} = [y_1, y_2, \dots, T_y].$$

**Table 2.** Percentage distribution of similarity between 5W1H and PropBank in our dataset

PropBank Role	Who	What	When	Where	Why	How
ARG0	<b>84.48</b>	0.00	3.33	0.00	0.00	0.00
ARG1	10.34	<b>53.85</b>	0.00	0.00	0.00	0.00
ARG2	0.00	9.89	0.00	0.00	0.00	0.00
ARG3	0.00	0.00	0.00	22.86	0.00	0.00
ARG4	0.00	3.29	0.00	34.29	0.00	0.00
ARGM-TMP	0.00	1.09	<b>60.00</b>	0.00	0.00	0.00
ARGM-LOC	0.00	1.09	10.00	<b>25.71</b>	0.00	0.00
ARGM-CAU	0.00	0.00	0.00	0.00	<b>100.00</b>	0.00
ARGM-ADV	0.00	4.39	20.00	0.00	0.00	0.06
ARGM-MNR	0.00	3.85	0.00	8.57	0.00	<b>90.91</b>
ARGM-MOD	0.00	4.39	0.00	0.00	0.00	0.00
ARGM-DIR	0.00	0.01	0.00	5.71	0.00	3.03
ARGM-DIS	0.00	1.65	0.00	0.00	0.00	0.00
ARGM-NEG	0.00	1.09	0.00	0.00	0.00	0.00

An *encoder* state is represented by the concatenation of the hidden states:

$$\mathbf{h}_i = \left[ \overrightarrow{h_i} : \overleftarrow{h_i} \right]^T, \quad i = 1, \dots, T_x. \quad (2)$$

The attention mechanism plugs a context vector  $c_t$  between the encoder and the decoder. For each single word that the decoder wants to generate, the context vector is used to compute the probability distribution of source language words.

The context vector  $c_t$  depends on a sequence  $(h_1, \dots, h_{T_x})$  to which an encoder maps the input sentence. The decoder has hidden states  $s_t = f(s_{t-1}, y_{t-1}, c_t)$  for the output word at position  $t$ ,  $t = 1, \dots, m$ , where the context vector  $c_t$  is a sum of hidden states of the input sequence, weighted by alignment scores:

$$c_t = \sum_{i=1}^n \alpha_{t,i} \mathbf{h}_i, \quad (3)$$

$$\alpha_{t,i} = \frac{\exp(\text{score}(s_{t-1}, \mathbf{h}_i))}{\sum_{i'=1}^n \exp(\text{score}(s_{t-1}, \mathbf{h}_{i'}))}, \quad (4)$$

$\alpha_{t,i}$  is the alignment score assigned to the pair of input at position  $i$  and output at position  $t$ ,  $(y_t, x_i)$  which depends on the proximity of their match. The extent of each source hidden state to be chosen for each output depends on the set of weights

$\{\alpha_{t,i}\}$ . To parameterize the alignment score  $\alpha$ , a feed-forward network with a single hidden layer is used and this network is jointly trained with other parts of the model. Therefore, the score function is represented in the following form, given that  $\tanh$  is used as the non-linear activation function:

$$\text{score}(s_t, \mathbf{h}_i) = \mathbf{v}_a^T \tanh(\mathbf{W}_a[s_t; \mathbf{h}_i]), \quad (5)$$

where both  $\mathbf{v}_a$  and  $\mathbf{W}_a$  are weight matrices to be learned in the alignment model.

## 4.2 Architecture

Our deep attention architecture is in similar lines with Bahdanau et. al. [1] where the input is a sequence of words  $(x_1, x_2, \dots, x_{T_x})$  consisting of source word tokens and 5W1H tokens. The input sequence are mapped to the target sequence  $(y_1, y_2, \dots, y_{T_y})$  by our trained model(s). The source sentence is represented as dense word vectors which is then fed to a Bi-directional LSTM encoder to generate the hidden states as stated in eq.(2). The information generated by the hidden states is then used by the decoder to output the target tokens recursively. The architecture of the deep neural network is shown in Fig 1.

### 4.2.1 Encoder

Our encoder is a B-directional RNNs (Recurrent Neural Network) with LSTM cells. The encoder

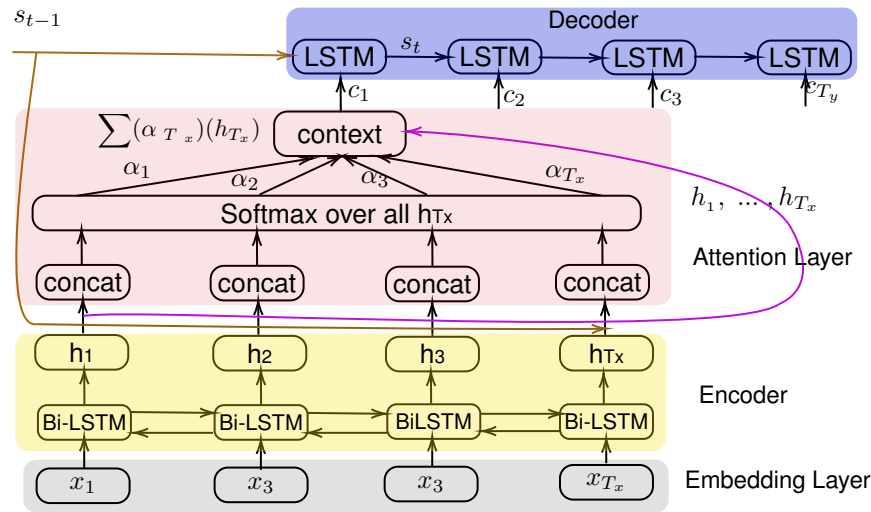


Fig. 1. Deep Neural Attention Network

outputs hidden states  $\left[ \begin{matrix} \rightarrow : \leftarrow \\ h_i & h_i \end{matrix} \right]$  where each  $h_i$  contains information about the surrounding context of the word  $x_i$ . We refer to  $\mathbf{M}$  as the complete matrix of encoder hidden states. Since the length of the input sequence is  $T_x$ , the shape of the output of the encoder is  $T_x \times 2\mathbf{M}$ .

#### 4.2.2 Attention

The attention mechanism is a feed-forward neural network of two layers. In the first layer, at every time step  $t$ , we use the concatenation of the forward and backward source hidden states  $h$  in the bi-directional encoder and target hidden states  $s_{t-1}$  in the previous time step of the non-stacking unidirectional decoder. The score  $score(s_t, h_i)$  in eq.(5) is the result of the concatenation which is then passed to a softmax to generate the  $\alpha_{t,i}$ . Since the  $\alpha_{t,i}$  is generated for only one hidden state  $h_i$ , we need to apply the softmax over all the  $h$  of the input sequence  $T_x$ . This is obtained by copying the  $s_{t-1}$  to all the  $h$  in  $T_x$ :

$$\begin{aligned} \text{out}(1) &= \text{NeuralNet}([s(t-1), h(1)]), \\ \text{out}(2) &= \text{NeuralNet}([s(t-1), h(2)]), \\ &\dots \\ \text{out}(T_x) &= \text{NeuralNet}([s(t-1), h(T_x)]), \end{aligned}$$

Therefore,  $\alpha_{t,i}$  is calculated as:

$$\alpha_{t,i} = \frac{\exp(\text{score}(s_{t-1}, h_i))}{\sum_{i=1}^{T_x} \exp(\text{score}(s_{t-1}, h_{T_x}))}. \quad (6)$$

The generated  $\alpha_{t,i}$  are then used with the hidden states  $h$  in  $T_x$  to compute the context vector  $c_t$  which is a weighted sum of the products of  $\alpha_{t,i}$  and  $h_i$  in  $T_x$  as shown in eq.(3).

#### 4.2.3 Decoder

The decoder is a single layer unidirectional LSTM network responsible for generating the output token  $y_t$  where  $t = 1, 2, \dots, T_y$ . A learned distribution over the vocabulary at each time step is used to generate  $y_t$  from  $t$  given its state  $s_t$ , the previous output token  $y_{t-1}$ , the context vector  $c_t$  and  $\mathbf{M}$ . We can parameterize the probability of decoding each word  $y_t$  as:

$$p(y_t | s_t, y_{t-1}, c_t, \mathbf{M}) = \text{softmax}(g(s_t)), \quad (7)$$

where  $g$  is the transformation function that outputs a vocabulary-sized vector. Here,  $s_t$  is the RNN hidden unit, abstractly computed as:

$$s_t = f(s_{t-1}, h), \quad (8)$$

where  $f$  computes the current hidden state given the previous hidden state in a LSTM unit.



Minimization of the negative log-likelihood of the target token  $y_t$  for each time step is the primary objective of our model at the time of training. The loss for the whole sequence ( $X$ ) is calculated as:

$$loss = -\frac{1}{T_y} \sum_{t=0}^{T_y} \log P(y_t | y < t, X). \quad (9)$$

## 5 Experiments

### 5.1 Dataset

We used two different datasets, one based on the US Elections held in November, 2016 and the other based on the #MeToo<sup>4</sup> campaign. The dataset on the US Elections are taken from [30] containing 3000 English tweets. For the #MeToo dataset, we crawled 248,160 tweets using hash tags such as #MeToo, #MeTooCampaign, #MeTooControversy, #MeTooIndia, as query with the *twitter4j*<sup>5</sup> API. We applied regular expressions to remove the Re-tweets (tweets with RT prefix) and Non-English tweets. Most of the Non-English tweets are in Roman transliterated form and therefore, they are manually removed.

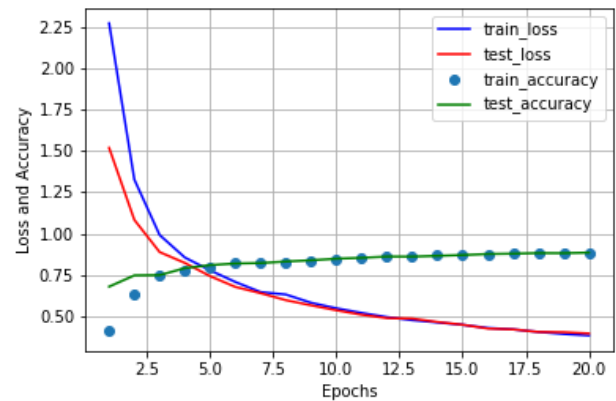
After manually removing the re-tweets and Non-English tweets, the dataset is finally reduced to 8175 tweets. The reason for such a drastic reduction is due to the presence of tweets containing Roman transliterated Non-English words. All the tweets are then tokenized with CMU tokenizer [12]. We prepared the datasets in such a manner that for every tweet that has multiple predicates, the tweet is repeated in the corpus for each predicate (Table 3).

### 5.2 Model Setup

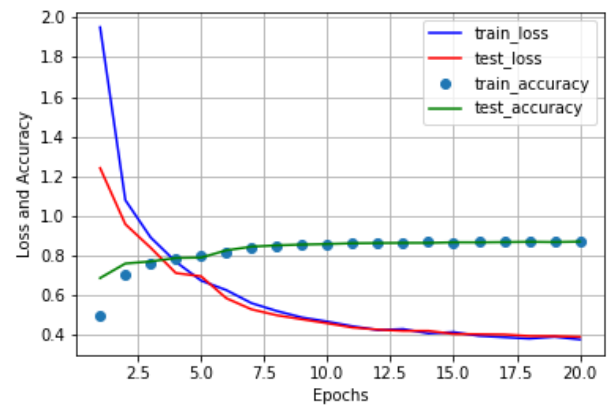
We setup our model with Keras [8] and initialize the model with pre-trained 300-dimensional GloVe [24] embeddings. Our vocabulary size is set to  $|\nu| \approx 40K$  words with a maximum sequence length = 100. Both the *encoder* and *decoder* are set with latent dimensions of 256 respectively. For the *attention* layer, we use Keras RepeatVector [8]

<sup>4</sup>[https://en.wikipedia.org/wiki/Me\\_Too\\_movement\\_\(India\)](https://en.wikipedia.org/wiki/Me_Too_movement_(India))

<sup>5</sup><http://twitter4j.org/en/index.html>



(a) Loss and Accuracy on the US elections dataset



(b) Loss and Accuracy on the #MeToo dataset

Fig. 2. Reported loss and accuracy

set to the maximum length of the input sequence. The concatenation  $[s_{t-1}, h]$  is obtained by merging them into a concatenated layer. The *attention* layer is implemented with two *dense* layers set with *tanh* and *softmax* respectively. The softmax is applied across all the hidden states from the encoder. For obtaining the context  $c_t$ , at every time step  $t$ , a different layer is required. We implemented this using Lambda layer wrappers of the Keras API.

### 5.3 Learning

Our models are created on a fifth generation Intel core i7 based machine with four cores and 16

**Table 3.** Dataset structure. Every tweet that has  $n$  predicates is repeated  $n$  times in the dataset

Repeat count	predicate	Tweet
1	attempt	[Apple CEO Tim Cook] <sub>WHO</sub> [attempts] <sub>V</sub> [to unify staff in wake of Trump victory] <sub>WHAT</sub>
2	unify	[Apple CEO Tim Cook] <sub>Who</sub> attempts to [unify] <sub>V</sub> [staff in wake of Trump victory] <sub>WHAT</sub>

gigabytes of Random Access Memory without any GPU (Graphics Processor Unit) support. Due to the lack of a GPU support, a single epoch takes a considerable amount of time. With the available system configuration, a single epoch took around 30 to 40 minutes. We therefore, experimented with only different epochs of 5, 10 and 20 and got the best results with 20 epochs with a batch size,  $BATCH\_SIZE = 1000$ . We use Adam optimizer [15] and a learning rate  $l_r = 0.1$ . The dataset was split into 90% train and 10% test sets.

## 6 Results

The objective of our work is to extract the 5W1H from tweets. But for comparison with previous [30] SRL systems on tweets, we evaluated our system (*Deep-SRL*) for PropBank role identification task. In Table 4, we give the comparison of our system (*Deep-SRL*) with the SRL system of Rudrapal et. al. [30] (*DRP-SRL*) on the PropBank role identification task. For evaluation purpose, we used the standard measures of *Precision*, *Recall* and *F-1*.

The comparison is done on the *US Elections 2016* dataset, on which our system outperformed *DRP-SRL* system by overall F-1 of 28.72. This is a significant improvement over previous results. In Table 5, we give the performance of *Deep-SRL* for 5W1H extraction on both the two datasets (*US Elections 2016* and *metoo movement*). *Deep-SRL* achieves an overall F-1 score of 88.21 in the whole corpus.

Fig 2(a) and (b) show the loss and accuracy of our model on the train and test sets on both the datasets respectively. The reported loss at *epoch* = 20 for the *US Elections* dataset is 0.5 and that for

the *#MeToo* dataset is 0.45. This indicates a drop in the loss by 0.05.

Our models reported an accuracy of 88.32% for the *US Elections* dataset and 88.15% for the *#MeToo* dataset. The three metrics of precision, recall and F-1 score is shown in fig 3(a) and (b).

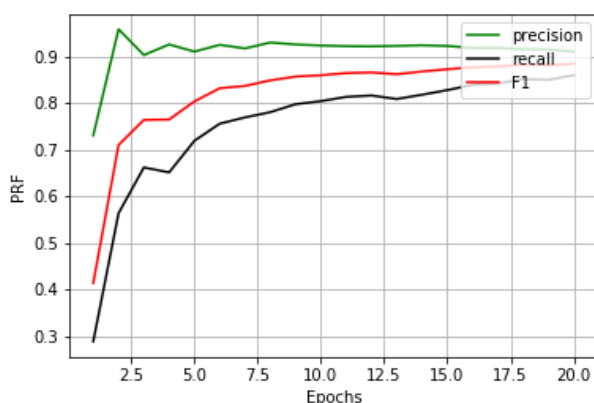
## 7 Analysis

Since we adopted the BIO<sup>6</sup> tagging format, it is necessary to identify the **argument span**. Here, argument span means the maximum number of tokens falling under a WHO or WHAT or WHEN or WHERE or WHY or HOW. To verify argument spans, we measure the percentage of overlap between the predicted argument spans and the gold spans. We found that 85.4% of the predicted spans match the gold spans completely, 5.23% of the predicted spans are partially overlapping with gold spans, and 9.37% of the predicted spans do not overlap at all with gold. There are partial overlaps because the model could not tag some of the group of tokens with a proper BIO sequence. For example, a token which is supposed to be tagged with a B-WHO, was tagged as I-WHO.

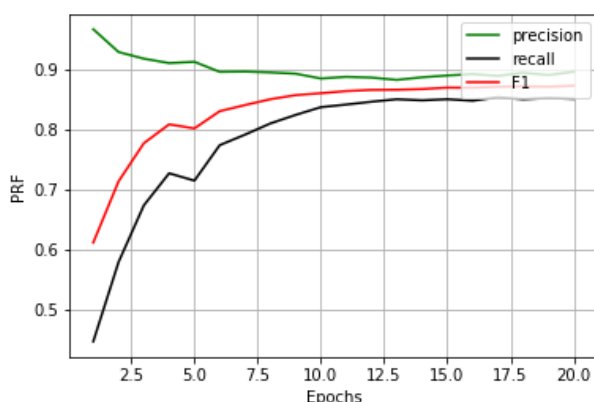
**Table 4.** Comparison of *DRP* and our system on the PropBank role identification task for the *US Election* corpus.

System	#Tweets	F-1
DRP-SRL	3000	59.76
Deep-SRL	3000	<b>88.48</b>

<sup>6</sup>[https://en.wikipedia.org/wiki/Inside%E2%80%93beginning\\_\(tagging\)](https://en.wikipedia.org/wiki/Inside%E2%80%93beginning_(tagging))



(a) Precision, Recall and F-1 on the US elections dataset



(b) Precision, Recall and F-1 on the #MeToo dataset

Fig. 3. Model performance on all the datasets

## 8 Conclusion

SRL based on the PropBank role set is a fine-grained approach but requires deep knowledge about the role arguments for annotation. In contrast, our simplistic 5W1H approach is easier to annotate a corpus with a little compromise at the fine-grain level role set identification task. In this work, we proposed a deep attention based neural network for the task of semantic role labeling by extracting the 5W1H from tweets. We trained our models and evaluated them on the 2016 US Elections dataset that was used by a previous SRL system for tweets. We compared

**Table 5.** Our System (Deep-SRL) for 5W1H extraction on both the US Election and #MeToo corpus.

Corpus	Precision	Recall	F-1
US Elections	90.87	86.21	88.48
#MeToo	90.63	85.40	87.94
Average	90.75	85.8	<b>88.21</b>

our models with previous SRL systems on tweets and observed a significant improvement over the previous implementations. We also prepared a new dataset based on the #MeToo campaign and evaluated our models on them. Our experimental results indicate that our models substantially improve SRL performances on tweets. However, there are certain limitations in the 5W1H adoption as the fine-grain semantic roles are ignored in such an approach, thus, limiting the in-depth SRL role identification. However, the 5W1H concept could be very convenient to perform other information extraction tasks such as event detection, even summarization on tweets.

## References

1. Bahdanau, D., Cho, K., & Bengio, Y. (2015). Neural machine translation by jointly learning to align and translate. *Proceedings of the ICLR Conference.*, ICLR, pp. .
2. Baker, C. F., Fillmore, C. J., & Lowe, J. B. (1998). The Berkeley FrameNet project. *Association for Computational Linguistics '98 Proceedings of the 36th Annual Meeting of the Association for Computational Linguistics and 17th International Conference on Computational Linguistics*, Vol. 1, pp. 86–90.
3. Bastianelli, E., Castellucci, G., Croce, D., & Basili, R. (2013). Textual inference and meaning representation in human robot interaction. *Proceedings of the Joint Symposium on Semantic Processing. Textual Inference and Structures in Corpora*, JSSP.
4. Bharati, A., Chaitanya, V., & Sangal, R. (1994). *Natural Language Processing a Paninian Perspective*, volume 1. Prentice-Hall, PHI, New Delhi.

5. **Carreras, X. & Màrquez, L. (2005).** Introduction to the conll-2005 shared task: semantic role labeling. *Proceedings of the Ninth Conference on Computational Natural Language Learning*, Association for Computational Linguistics, pp. 152–164.
6. **Chakma, K. & Das, A. (2018).** A 5w1h based annotation scheme for semantic role labeling of english tweets. *Computación y Sistemas*, Vol. 22, No. 3, pp. 747–755.
7. **Cheng, J., Dong, L., & Lapata, M. (2016).** Long short-term memory-networks for machine reading. *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing*, Association for Computational Linguistics, pp. 551–561.
8. **Chollet, F. et al. (2015).** Keras. <https://keras.io>.
9. **Dan, S. & Lapata, M. (2007).** Using semantic roles to improve question answering. *Proceedings of the Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning*, Association for Computational Linguistics, pp. 12–21.
10. **Genest, P. & Lapalme, G. (2011).** Framework for abstractive summarization using text-to-text generation. *Proceedings of the Workshop on Monolingual Text-To-Text Generation*, Association for Computational Linguistics, pp. 64–73.
11. **Gildea, D. & Jurafsky, D. (2002).** Automatic labeling of semantic roles. *Association for Computational Linguistics*, Vol. 28, No. 3, pp. 245–288.
12. **Gimpel, K., Schneider, N., O'Connor, B., Das, D., Mills, D., Eisenstein, J., Heilman, M., Yogatama, D., Flanigan, J., & Smith, N. A. (2011).** Part-of-speech tagging for twitter: Annotation, features, and experiments. *Proceedings of Association for Computational Linguistics 2011*, Association for Computational Linguistics, pp. 42–47.
13. **He, L., Lee, K., Lewis, M., & Zettlemoyer, L. S. (2017).** Deep semantic role labeling: What works and what's next. *Association for Computational Linguistics*, pp. 473–483.
14. **Hochreiter, S. & Schmidhuber, J. (1997).** Long short-term memory. *Neural Computation*, Vol. 9, No. 8, pp. 1735–1780.
15. **Kingma, D. P. & Ba, J. (2014).** Adam: A method for stochastic optimization. *CoRR*, Vol. abs/1412.6980.
16. **Knight, K. & Luk, S. K. (1994).** Building a large-scale knowledge base for machine translation. *AAAI, American Association for Artificial Intelligence*, pp. 773–778.
17. **Liu, X., Li, K., Han, B., Zhou, M., Jiang, L., Xiong, Z., & Huang, C. (2010).** Semantic role labeling for news tweets. *Proceedings of the 23rd International Conference on Computational Linguistics (Coling 2010)*, Association for Computational Linguistics, pp. 698–706.
18. **Liu, X., Li, K., Han, B., Zhou, M., & Xiong, Z. (2011).** Collective semantic role labeling for tweets with clustering. *Proceedings of the Twenty-Second international joint conference on Artificial Intelligence (IJCAI'11)*, Vol. 3, pp. 1832–1837.
19. **Marcheggiani, D. & Titov, I. (2017).** Encoding sentences with graph convolutional networks for semantic role labeling. *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*, Association for Computational Linguistics, pp. 1506–1515.
20. **Mohammad, S. M., Zhu, X., & Martin, J. (2010).** Semantic role labeling of emotions in tweets. *Proceedings of the 5th Workshop on Computational Approaches to Subjectivity, Sentiment and Social Media Analysis*, pp. 32–41.
21. **Moschitti, A., Morarescu, P., & Harabagiu, S. M. (2003).** Open domain information extraction via automatic semantic labeling. *FLAIRS Conference*, pp. 397–401.
22. **Palmer, M., Gildea, D., & Kingsbury, P. (2005).** The proposition bank: A corpus annotated with semantic roles. *Computational Linguistics Journal*, Vol. 31, No. 1, pp. 71–105.
23. **Palmer, M., Gildea, D., & Xue, N. (2010).** Semantic role labeling. *Synthesis Lectures on Human Language Technology Series*, Morgan and Claypool.
24. **Pennington, J., Socher, R., & Manning, C. D. (2014).** Glove: Global vectors for word representation. *EMNLP*, pp. 1532–1543.
25. **Platt, J. C. (1998).** Sequential minimal optimization: A fast algorithm for training support vector machines. Technical report, ADVANCES IN KERNEL METHODS - SUPPORT VECTOR LEARNING.
26. **Pradhan, S., Haciogl, K., Ward, W., Martin, K., & Jurafsky, D. (2005).** Semantic role chunking combining complimentary syntactic views. *Proceedings of the 9th Conference on Computational Natural Language Learning*, Association for Computational Linguistics, pp. 217–220.

27. **Punyakanok, V., Roth, D., & Yih, W.-T. (2005).** The importance of syntactic parsing and inference in semantic role labeling. *Computational Linguistics*, Association for Computational Linguistics, pp. 257–287.
28. **Robertson, D. & Bruce, W. (1946).** A note on the classical origin of 'circumstances' in the medieval confessional. *Studies in Philology*, Vol. 43, No. 1, pp. 6–14.
29. **Roth, M. & Lapata, M. (2016).** Neural semantic role labeling with dependency path embeddings. *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics*, Association for Computational Linguistics, pp. 1192–1202.
30. **Rudrapal, D. & Das, A. (2018).** Semantic role labelling of english tweets through sentence boundary detection. *International Journal of Intelligent Information and Database Systems*, Vol. 11, No. 4, pp. 225–235.
31. **Schuler, K. K. & Palmer, M. S. (2005).** *Verbnet: a broad-coverage, comprehensive verb lexicon*. Ph.D. thesis, University of Pennsylvania Philadelphia, PA, USA, Philadelphia, USA.
32. **Strubell, E., Verga, P., Andor, D., Weiss, D., & McCallum, A. (2018).** Linguistically-informed self-attention for semantic role labeling. *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, Association for Computational Linguistics, pp. 5027–5038.
33. **Surdeanu, M., Harabagiu, S., Williams, J., & Aarseth, P. (2003).** Using predicate-argument structures for information extraction. *Proceedings of the 41st Annual Meeting on Association for Computational Linguistics*, Association for Computational Linguistics, pp. 8–15.
34. **Sutskever, I., Vinyals, O., Le, Q. V., & McCallum, A. (2014).** Sequence to sequence learning with neural networks. *Proceedings of NIPS*, MIT Press, pp. 3104–3112.
35. **Tan, Z., Wang, M., Xie, J., Chen, Y., & Shi, X. (2018).** Deep semantic role labeling with self-attention. *Proceedings of the Thirty-Second AAAI Conference on Artificial Intelligence, (AAAI-18), the 30th innovative Applications of Artificial Intelligence (IAAI-18), and the 8th AAAI Symposium on Educational Advances in Artificial Intelligence (EAAI-18)*, pp. 4929–4936.
36. **Ueffing, N., Haffari, G., & Sarkar, A. (2007).** Transductive learning for statistical machine translation. *Proceedings of the 45th Annual Meeting of the Association of Computational Linguistics*, Association of Computational Linguistics, pp. 25–32.
37. **Wilson, T. & Medine, P. (1994).** *The Art of Rhetoric (1560)*. G - Reference, Information and Interdisciplinary Subjects Series. Pennsylvania State University Press.
38. **Wu, D. & Fung, P. (2009).** Semantic roles for smt: a hybrid two-pass model. *Proceedings of Human Language Technologies*, Association for Computational Linguistics, pp. 13–16.
39. **Zhao, Z., Sun, J., Mao, Z., Feng, S., & Bao, Y. (2016).** Determining the topic hashtags for Chinese microblogs based on 5W model. *Proc. of the second International Conference, BigCom 2016*, Springer, pp. 55–67.
40. **Zhou, J. & Xu, W. (2015).** End-to-end learning of semantic role labeling using recurrent neural networks. *Proc. of the Annual Meeting of the Association for Computational Linguistics*, Association for Computational Linguistics, pp. 1127–1137.

Article received on 23/01/2019; accepted on 04/03/2019.  
Corresponding author is K. Chakma.



# English Dataset For Automatic Forum Extraction

Jakub Sido, Miloslav Konopík, Ondřej Pražák

University of West Bohemia, Faculty of Applied Sciences,  
NTIS – New Technologies for the Information Society,  
Czech Republic

{sidoj, konopik, ondfa}@kiv.zcu.cz,

**Abstract.** This paper describes the process of collecting, maintaining and exploiting an English dataset of web discussions. The dataset consists of many web discussions with hand-annotated posts in the context of a tree structure of a web page. Each post consists of username, date, text, and citations used by its author. The dataset contains 79 different websites with at least 500 pages from each. Each web page consists of a tree structure of HTML tags with texts taken from selected web pages. In the paper, we also describe algorithms trained on the dataset. The algorithms employ basic architectures (such as a bag of words with an SVM classifier and an LSTM network) to set a baseline for the dataset.

**Keywords.** Information retrieval, web discussion.

## 1 Introduction

In the last years, a significant portion of human social lives moved into the Internet, and many social networks have appeared since. Long ago, it became clear that these networks contain a lot of valuable information.

Nowadays, the most significant social networks are heavily monitored and analyzed by various autonomous algorithms. For big social networks, it makes sense to create a dedicated crawling script to gather the data it contains. However, small networks appear and disappear on a daily basis, and such effort is not profitable. Nevertheless, a lot of valuable information may be gathered when all the small networks are combined.

Every social network is unique to some extent. Crawling scripts would have to be manually created (or at least customized) for every network. Such

an effort is not very profitable for small networks. Therefore, we propose to create an automated extraction algorithm instead. In this paper, we present a dataset dedicated to training such algorithms.

Our primary motivation to target small social networks consists of monitoring potentially harmful, unwanted or dangerous activities on the web. This would allow early prevention of such activities (suicides, crimes against society, sexual abuse cases). We believe that small networks are very prone to bad activities in general because they are not currently so well monitored.

In the paper, we describe algorithms trained on the dataset. The algorithms employ basic architectures (such as a bag of words with an SVM classifier and an LSTM network) to set a baseline for the dataset. Even with these simple architectures, we reach a promising accuracy of data extraction. Still, we believe that there is a big room for further improvement.

## 2 Related Work

There are a few other works which are related to this topic. We can split them into two categories: datasets and tools. The first group deals with automatic data extraction from diverse sources. However, there are also some systems that operate on the forum data, too.

## 2.1 Datasets

Several datasets appeared in the last decade with increasing interest in the automatic data extraction. The datasets exist primarily for English. However, there are some datasets for other languages as well.

### 2.1.1 Automatic Extraction of Informative Block from webpages

This dataset was created in 2005. A group from the Pennsylvania State University took 11 news websites and random news from every section. Altogether, it was 5911 pages. This dataset is not publicly available[1].

### 2.1.2 Shallow Information Extraction from Medical Forum Data

This project[4] deals with searching for answers on the medical discussion forum. The authors took 175 posts in 50 threads from the Healthboards forum. The dataset is publicly available<sup>1</sup>.

### 2.1.3 Exploiting thread structures to improve smoothing of language models for forum post retrieval

This project [2] used a dataset created by the authors from the CNET "Computer Help" forum for searching for relevant posts in the context of a query. They made the hard copy of 29 413 threads with 135 752 posts.

### 2.1.4 Learning Online Discussion Structures by Conditional Random Fields

This work[5] deals with searching relevant answers on discussion forums. The authors used a dataset they collected from three different online discussion forums. Altogether it contains more than 180 000 posts in 31838 threads. The dataset is publicly available.<sup>2</sup>

<sup>1</sup><http://sifaka.cs.uiuc.edu/ir/index.html>

<sup>2</sup><http://sifaka.cs.uiuc.edu/textasciitildewang296/Data/index.html>

## 2.2 Existing Systems

Several systems allow users to search through web discussions:

- <https://webhose.io>,
- <http://boardreader.com>,
- <http://omgili.com/>.

Many of them are operated commercially. Some of them offer free usage with a restricted number of queries. However, none of them provides raw data for any further analysis.

## 3 Dataset Design

### 3.1 Target Language

The presented dataset consists of websites in the English language. In the future, we will extend the dataset to other languages.

### 3.2 Annotation process

During the annotation process, the annotators downloaded the web pages with discussions and labelled by hand using XPath. For each forum, they created manually the XPaths which specifies the informative field's (e. g. author names, dates, and texts) position in an HTML tree. Annotator could use XPath extraction tool (for example developer tools in a web browser), but XPath had to be generalized by hand. The task of the annotators was to define an XPath for each class for every annotated site.

Every useful part of the post is classified into one of six classes:

- *author*,
- *date*,
- *text*,
- *citation author*,
- *citation date*,
- *citation text*.



In order to make evaluations of the prospective future systems more accurate, we labelled only those elements that contain the desired text information<sup>3</sup> directly. It is, of course, possible to mark whole subtrees of these tags as relevant but then we could get potentially textless tags as class-labeled.

You can see the annotation output example on Figure 1.

Afterwards, every web page is automatically transformed into a representation in an easily machine-readable format for further processing. We will describe the transformation later in the section 4. This format preserves the tree structure of a web page but makes further analysis much easier. However, the original pages with XPath paths are available, too.

### 3.3 Design Decisions

Many forum pages do not conform to any HTML markup standard (they are not valid). In some other cases, they are valid, but they introduce other problems. For example, the information is not present in any structure on some forums. This means that all the data is written in one HTML element. We also have to deal with citations, data deletion and other issues. In the following sections, we describe our solution to these problems.

**Citations And Responses** Citations or responses are used often on the web forums. As the original idea of creating this dataset is to be able to detect and extract posts, we do not try to keep the tree of interactions between users as the authors of [1] do. We want to mark all users and their posts. Thus, every post was grabbed regardless of whether it is a reply. At the same time, this structure can be derived from the tree structure of the web page, which we include in the dataset as well.

Example of response:

```
John : Is anybody interested in ...
Luke : Yes, I am.
Pete : I am not.
```

Only the citations which are often used directly in users' texts should be separated from the posts. In some cases, the citation does not contain the whole content or the date of the original post. Thus, it is necessary to create new classes for them. Example of citation:

```
Luke : "Is anybody interested
in ..." (John)
Yes, I am !
```

**Template Information** On some forums, posts are marked as deleted instead and they are still present in the HTML tree. Usually, the date of publication of such post stays in its position, but the text is replaced with some template text. In some cases, the original element stays on its position, and only the text is substituted for example with one of the following:

- This post was deleted by the author.
- Deleted by admin one day ago.

Sometimes the original subtree with the post is replaced with another one e.g.

```
<div class="post">
  <h4>John</h4>
  <p>Text of john's post</p>
</div>
```

is replaced by:

<sup>3</sup>E.g., post text or nickname of an author.

```
{
  "author": "//a[@class='bigusername']",
  "date": "//td/div[@class='normal'][2]",
  "text": "//tr/td[@class='alt1']/div[1]",
  "citation_author": "//tr/td[@class='alt2']/div[1]",
  "citation_date": "/NONE",
  "citation_text": "//tr/td[@class='alt2']/div[2]"
}
```

**Fig. 1.** Annotation output example

```
<div class="deleted">
  <span>Deleted</span>
</div>
```

In both cases, it is up to the prospective user of this data set to handle it.

**Missing Data** Citations are mentioned with the dates of original publication on some forums, while on other forums they are not. For this reason, it is possible that the information about the date of publication of the cited post is missing.

**Mixed elements** During creating of the data set, we found several forums that have different pieces of semantic information mixed within one element. In most cases, it was the date of publication and the author's name that was mixed. Like:

```
<div>
  Posted by John one day ago
</div>
```

Considering the fact we want to keep the data set as simple as possible and to the number of such forums, we decided not to include these sites in the data set.

**Table 1.** Mapping of classes on id number

Class	Id
other - everything other like ads, e.g.	0
nick - a username used by an author	1
date - a date of publication of the post	2
text - a text of the published post	3
citation author - a username of the cited author	4
citation date - a date of the original cited post	5
citation text - a text of the citation	6

## 4 Data Format

We designed the dataset taking into account the tree structure of the web pages. It is naturally possible to walk through the tree-structured page using the standard graph algorithms. The elements in the data set files are ordered as the preorder search finds them.. Each element contains the id of its parent to keep the tree structure.

Each node is represented by a separate line in the format:

```
ID PARENT_ID TAG_NAME CLASS TEXT
```

Where class is defined by mapping in Table 1.

Let's suppose we have a structure like this:

**Table 2.** Distribution of classes

Class	Relative frequency
nick	1.8%
date	1.8%
text	2.2%
citation author	1.9%
citation date	0.8%
citation text	1.5%
others	90%

**Table 3.** Statistics of counts of downloaded pages

Total number of forums	79
– training part	50
– testing part	29
Total number of pages	65 242
Minimum pages per forum	501
Maximum pages per forum	2317
Average number of pages per forum	826

## 5 Classes Counts And Statistics

The Table 2 shows some basic dataset statistics.

We can notice the imbalance between the class "other" and the remaining classes. However, it is an expected result provided that the number of tags with the target information is very low.

## 6 First Experiments

In order to set a baseline on the dataset, we conduct first experiments of data extraction from the forums. The experiments consist of *preprocessing*, *feature extraction* and *classification*. We describe all the steps in the following sections.

### 6.1 Preprocessing

The forums have some specific properties regarding the employed language and vocabulary. For example, author's names (nicks) are typically composed of a mixture of (usually artificial) names, numbers and special characters. Such words create a lot of low-frequency vocabulary items that are hard to classify. Therefore, we transform some specific groups of characters into predefined symbols. The groups of our interest are:

- Capital letters,
- Small letters,
- Numbers,
- Non-alphanumeric characters.

```
<div>
  <div>
    <p>Posted by John</p>
  </div>
  <div>
    <span>
      One day ago
    </span>
    This is my favourite place
    in Roma.
    <img />
    On Saturday I'm flying to
    Italy.
  </div>
</div>
```

This structure will be transformed into the following format:

```
0;-1;div;0
1;0;div 0
2;1;p;1;Posted by John
3;0;div;3;This is my favourite place in Roma.
  On Saturday I'm flying to Italy.
4;3;span;2;One day ago
5;3;img;0
```

In this, way the dataset is converted into the format suitable for other processing.

**Table 4.** Example of replacing groups of characters

Infrequent words	Mapped on
Jack59 Frank41 Stephan235	Aa1
john.23 george-4	a-1
2012-04-03 2009-03-12	1-1-1

Thanks to the above-described transformations, usernames like *Jack59*, *Frank41* or *Stephan235* are projected onto the same word representation for the subsequent processing. The same happens with the dates like *2013-08-04* and *2001-02-07*. Other examples can be found in Table 4.

Next, we use other standard preprocessing techniques such as tokenization based on a regular expression and lower casing. The lower casing is performed after the above-mentioned transformation.

## 6.2 Classification Classes

Our dataset contains some classes (see Table 1) that are relatively similar. The similar classes form the following pairs: *nick* – *citation author*, *date* – *citation date*, *text* – *citation text*.

We expect that these pairs would create problems for the classifier. Therefore, we have decided to merge the pairs of similar classes. In our results, we show scores for both reduced four classes dataset and the original seven classes dataset.

## 6.3 Features

In order to keep our classification architectures simple and straightforward, we use the following basic features:

- *K*-most frequent words (the *K* is depended on the classifier – see section 6.4).
- Character masks – created by the transformation described in section 6.1.
- HTML tags (67 different HTML tags such as *div*, *img*, *p* etc.).

**Table 5.** 4 class classification - authors, dates and texts classes are merged see 6.2

Classifier	Text + Mask	HTML Tags	F1
SVM	✓	✗	47.45
	✗	✓	36.52
	✓	✓	54.92
LSTM	✓	✗	62.74
	✓	✓	<b>65.17</b>

**Table 6.** 7 class classification

Classifier	Text + Mask	HTML Tags	F1
SVM	✓	✗	34.28
	✗	✓	27.72
	✓	✓	40.36
LSTM	✗	✓	45.35
	✓	✓	<b>48.14</b>

## 6.4 Classifiers

We employ two classifiers: the SVM classifier and the LSTM [3] classifier.

The *SVM classifier* uses bag-of-words features based on a dictionary created from 200 most frequent words and 530 masks (section 6.1) from the training part of the dataset. HTML tags are in the form of one hot vector.

The *LSTM classifier* is a recurrent neural network that takes sequences of words as input. In our approach, we use randomly initialized embeddings to convert words into low dimensional vectors of real numbers that are fed on the input of the LSTM network. In this approach, we consider words that occur at least 15 times in the training portion of the dataset and the same set of 530 masks. We use the following hyper-parameters: *dimension of word embeddings*: 300, *LSTM hidden dimension*: 256, *dropout rate*: 0.5, *learning rate*: 0.0001, *optimization algorithm*: Adam.

## 6.5 Experiment Results

Following tables summaries the results of the different configurations of experiments.

The dataset divided to train, test and validate parts will be freely available on our website as well as the implementation of the experiment.

## 7 Future Work

**Dataset Extension** We plan to extend the dataset by adding more web-pages and different languages. With a bigger dataset and better algorithms (see the next paragraph) we intend to increase the dataset automatically. First, we will run the automatic extraction algorithm on a set of new web-pages. Then, we will generalize the XPath of the extracted elements to generate correct XPath for the web-pages. This process would have to be supervised at first, but we expect that at later phases, the whole process can be fully autonomous.

**Advanced Algorithms** A few issues showed up during the first experiments. Replacing of groups of characters brought some improvement, and when combined with the LSTM the results look promising. In the following research, we intend to focus on the tree structure of the web pages. The trees of web-pages are much larger than trees that parse trees. We expect that the task of dealing with the structure of web-pages will be fairly challenging.

Discussion contributions (posts) appear in a repetitive pattern on forums. We aim to design algorithms that would capture these patterns and use them to improve the classification accuracy.

## 8 Conclusion

The result of the above-described work is the new data set containing more than 30 000 web pages with forum discussions from 79 different web servers. This dataset can be used for training of algorithms for automatic extraction of forum posts from diverse sources. Concerning this purpose, it is designed to contain lots of different web servers with various layouts. This data set will be publicly accessible from our departmental web server.

Components of Advanced Technologies for the Pilsen Metropolitan Area (InteCom)"

## Acknowledgement

This work has been partly supported from ERDF "Research and Development of Intelligent (no.: CZ.02.1.01/0.0/0.0/17\_048/0007267) and by Grant No. SGS-2019-018 Processing of heterogeneous data and its specialized applications. Computational resources were provided by the CESNET LM2015042 and the CERIT Scientific Cloud LM2015085, provided under the programme "Projects of Large Research, Development, and Innovations Infrastructures"

## References

1. Debnath, S., Mitra, P., & Giles, C. L. (2005). Automatic extraction of informative blocks from webpages. In *Proceedings of the 2005 ACM Symposium on Applied Computing, SAC '05*. ACM, New York, NY, USA. ISBN 1-58113-964-0, 1722–1726. doi:10.1145/1066677.1067065.
2. Duan, H. & Zhai, C. (2011). Exploiting thread structures to improve smoothing of language models for forum post retrieval. In Clough, P., Foley, C., Gurrin, C., Jones, G. J. F., Kraaij, W., Lee, H., & Mudoch, V., editors, *Advances in Information Retrieval*. Springer Berlin Heidelberg, Berlin, Heidelberg. ISBN 978-3-642-20161-5, 350–361.
3. Hochreiter, S. & Schmidhuber, J. (1997). Lstm can solve hard long time lag problems. In *Advances in neural information processing systems*. 473–479.
4. Sondhi, P., Gupta, M., Zhai, C., & Hockenmaier, J. (2010). Shallow information extraction from medical forum data. In *Proceedings of the 23rd International Conference on Computational Linguistics: Posters, COLING '10*. Association for Computational Linguistics, Stroudsburg, PA, USA, 1158–1166.
5. Wang, H., Wang, C., Zhai, C., & Han, J. (2011). Learning online discussion structures by conditional random fields. In *Proceedings of the 34th International ACM SIGIR Conference on Research and Development in Information Retrieval, SIGIR '11*. ACM, New York, NY, USA. ISBN 978-1-4503-0757-4, 435–444. doi:10.1145/2009916.2009976.

Article received on 17/01/2019; accepted on 04/03/2019.  
Corresponding author is Jakub Sido.



# Enriching Word Embeddings with Global Information and Testing on Highly Inflected Language

Lukáš Svoboda<sup>1,2</sup>, Tomáš Brychcín<sup>1,2</sup>

<sup>1</sup> University of West Bohemia, Faculty of Applied Sciences,  
Department of Computer Science and Engineering,  
Czech Republic

<sup>2</sup> University of West Bohemia, Faculty of Applied Sciences,  
NTIS—New Technologies for the Information Society,  
Czech Republic

{svobikl,brychcin}@kiv.zcu.cz

**Abstract.** In this paper we evaluate our new approach based on the *Continuous Bag-of-Words* and *Skip-gram* models enriched with global context information on highly inflected Czech language and compare it with English results. As a source of information we use Wikipedia, where articles are organized in a hierarchy of categories. These categories provide useful topical information about each article. Both models are evaluated on standard word similarity and word analogy datasets. Proposed models outperform other word representation methods when similar size of training data is used. Model provide similar performance especially with methods trained on much larger datasets.

**Keywords.** Highly inflected language, word embeddings.

## 1 Introduction

The principle known as *Distributional hypothesis* is derived from the semantic theory of language usage, the meaning of words that are used and occur in the same contexts tend to have similar meaning [7]. The claim has the theoretical basics in psychology, linguistics, or lexicography [4]. This research area is often referred to as *distributional semantics*. During last years it has become a popular. Models based on this assumption are denoted as *distributional semantic models* (DSMs).

### 1.1 Distributional Semantic Models

DSMs learn contextual patterns from huge amount of textual data. They typically represent the meaning as a vector which reflects the contextual (distributional) information across the texts [35]. The words are associated with a vector of real numbers. Represented geometrically, the meaning is a point in a  $k$ -dimensional space. The words that are closely related in meaning tend to be closer in the space. This architecture is sometimes referred to as the *semantic space*. The vector representation allows us to measure similarity between the meanings (most often by the cosine of the angle between the corresponding vectors).

Word-based semantic spaces provide impressive performance in a variety of NLP tasks, such as language modeling [2], named entity recognition [14], sentiment analysis [11], and many others.

### 1.2 Local Versus Global Context

Different types of context induce different kinds of semantic spaces. [26] and [20] distinguish *context-word* and *context-region* approaches to the meaning extraction. In this paper we use the notation *local context* and *global context*, respectively. Global-context DSMs are usually based on *bag-of-words hypothesis*, assuming that the words are semantically similar if they occur in

similar articles and the order in which they occur in articles has no meaning. These models are able to register long-range dependencies among words and are more topically oriented.

In contrast, local-context *DSMs* collect short contexts around the word using moving window to induce the meaning. Resulting word representations are usually less topical and exhibit more functional similarity (they are often more syntactically oriented).

To create a proper *DSM* a large textual corpus is usually required. Very often Wikipedia is used for training, because it is currently the largest knowledge repository on the Web and is available in dozens of languages. The most of current *DSMs* learn the meaning representation merely from the word distributions and does not incorporate any metadata which Wikipedia offer.

### 1.3 Our Model

In [34] we show a different possibilities, how to incorporate global information and in this article we will summarize the work, choose ideal setup and test the model with highly inflected language. We combine both the local and the global context to improve the word meaning representation. We use local-context *DSMs* *Continuous Bag-of-Words* (CBOW) and Skip-Gram models [21].

We train our models on English and Czech Wikipedia. We evaluate it on standard word similarity and word analogy datasets.

### 1.4 Outline

The structure of article is following. Section 2 puts our work into the context of the state of the art. In Section 3 we review Word2Vec models on which our work is based. We define our model in Section 5 and 4. The experimental results presented in Section 7. We conclude in Section 8 and offer some directions for future work.

## 2 Related Work

In the past decades, simple frequency-based methods for deriving word meaning from raw text were popular, e.g. Hyperspace Analogue to Language [18] or Correlated Occurrence Analogue to Lexical Semantics [27] as a representatives of local-context *DSMs* and Latent Semantic Analysis [16] or Explicit Semantic Analysis [8] as a representatives of global-context *DSMs*. All these methods record word/context co-occurrence statistics into the one large matrix defining the semantic space.

Later on, these approaches have evolved in more sophisticated models. [21] revealed neural network based model *CBOW Skip-gram* that we are going to use as our baseline to incorporate Global context. His simple single-layer architecture is based on the inner product between two word vectors (detailed description is in Section 3). [25] introduced Global Vectors, the log-bilinear model that uses weighted least squares regression for estimating word vectors. The main concept of this model is the observation that global ratios of word/word co-occurrence probabilities have the potential for encoding meaning of words.

### 2.1 Local Context with Subword Information

Above mentioned models currently serve as a basis for many researches. [1] improved Skip-Gram model by incorporating a sub-word information. Similarly, in most recent study [30] incorporated a sub-word information into LexVec [29] vectors. This improvement is especially evident for languages with rich morphology. [17] used syntactic contexts automatically produced by dependency parse-trees to derive the word meaning. Their word representations are less topical and exhibit more functional similarity (they are more syntactically oriented).

[13] presented a new neural network architecture which learns word embeddings that capture the semantics of words by incorporating both local and global document context. It accounts for homonymy and polysemy by learning multiple embeddings per word.



Authors introduce a new dataset with human judgments on pairs of words in sentential context, and evaluate their model on it. Their approach is focusing on polysemous words and generally do not perform as well as Skip-Gram or CBOW.

### 3 Word2Vec

This section describes Word2Vec package which utilizes two neural network model architectures (CBOW and Skip-Gram) to produce a distributional representation of words [21]. Given the training corpus represented as a set of documents  $D$ . Each document (resp. article)  $a_j \in D$  is a sequence of words  $a_j = \{w_{j,i}\}_{i=1}^{L_j}$ , where  $L_j$  denote the length of the article  $a_j$ . Each word  $w$  in the vocabulary  $W$  is represented by two different vectors  $\mathbf{v}$  and  $\mathbf{u}$  depending whether it is used as a context word  $\mathbf{v}_w \in \mathbb{R}^d$  or a target word  $\mathbf{u}_w \in \mathbb{R}^d$ . The task is to estimate these vector representations in a way that optimize bellow described objective functions.

We use training procedure introduced in [22] called *negative sampling*. For the word at position  $i$  in the article  $a_j$  we define the negative log-likelihood:

$$E(w, \mathbf{h}) = -\log \sigma(\mathbf{u}_{w_o}^\top \mathbf{h}) - \sum_{w_n \in N} \log \sigma(\mathbf{u}_{w_n}^\top \mathbf{h}), \quad (1)$$

where  $N = (w_n \sim P(\mathbf{W}) | n = 1, \dots, K)$  is a set of negative samples (randomly selected words from a noise distribution  $P(\mathbf{W})$ ),  $w_o$  is the output word, and  $\mathbf{u}_{w_o}$  is its output vector;  $\mathbf{h}$  is the output value of the hidden layer:  $\mathbf{h} = \frac{1}{C} \sum_{c=1..N} \mathbf{v}_{w_c}$  for CBOW and  $\mathbf{h} = \mathbf{v}_{w_I}$  in the Skip-gram model;  $\sigma(x) = 1/(1 + \exp(-x))$ .

Considering articles  $a_j$ , in the *CBOW* architecture, the model predicts the current word  $w_{j,i}$  from a window of surrounding context words  $w_c \in C_{j,i}$ . The context is based on bag-of-words hypothesis, so that the order of the words does not influence the prediction. CBOW model optimizes following objective function:

$$\sum_{a_j \in D} \sum_{i=1}^{L_j} E(w_{j,i}, \frac{1}{|C_{j,i}|} \sum_{w_c \in C_{j,i}} \mathbf{v}_{w_c}). \quad (2)$$

According to [21], *CBOW* is faster than *Skip-Gram*, but *Skip-Gram* usually perform better for infrequent words.

### 4 Wikipedia Category Representation

Wikipedia is a good source of global information. Overall, Wikipedia comprises more than 40 million articles in 301 different languages. Each article references others that describe particular information in more detail. Wikipedia give more information about an article that we might not see at the first moment, such as mentioned links to other articles, or at the end of the article there is a section that describes all categories where current article is belonging. The category system of Wikipedia (see fig. 1) is organized as an overlapping tree [31] of categories<sup>1</sup> with one main category and a lot of subcategories. Every article contains several categories to which it belongs.

Categories are intended to group together with pages on similar subjects. Any category may branch into subcategories, and it is possible for a category to be a subcategory of more than one 'parent' category (A is said to be a parent category of B when B is a subcategory of A)[31]. The page editor uses either existing categories, or create one. Generally the user-defined categories are too vague or may not be suitable to use them in our model as a source of global information. Fortunately, Wikipedia provides 25 main topic classification categories for all Wikipedia pages.

For example the article entitled *Czech Republic* has categories *Central Europe*, *Central European countries*, *Eastern European countries*, *Member states of NATO*, *Member states of EU*, *Slavic countries and territories* and others.

Wikipedia categories provide very useful topical information about each article. In our work we use extracted categories to improve the performance of word embeddings. We denote articles as  $a_j$  and categories as  $x_k$  (see fig. 2).

<sup>1</sup><https://en.wikipedia.org/wiki/Portal:Contents/categories>

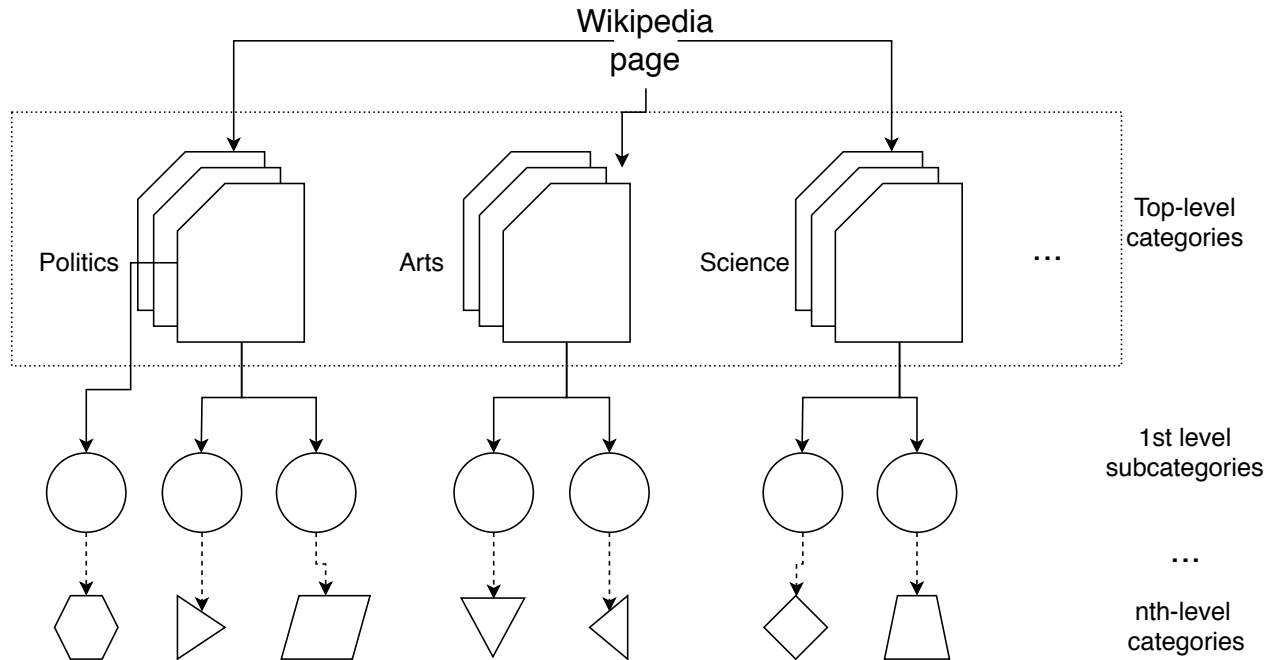


Fig. 1. Wikipedia category system

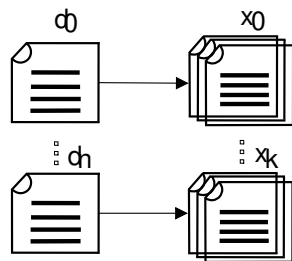


Fig. 2. Document - categories relation

## 5 Proposed Model

Some authors tried to extract a more concrete meaning using *Frege's principle of compositionality* [24], which states that the meaning of a sentence is determined as a composition of words. [36] introduced several techniques to combine word vectors into the final vector for a sentence. In [3] experimented with Semantic Textual Similarity, from the tests with words vector composition based on *CBOW* architecture, we can see that this method is powerful to carry the meaning of a sentence.

Our model is shown in Figure 3. We build up the model based on our previous knowledge and believes that Global information might improve the performance of word embeddings and further lead to improvements in many NLP subtasks.

Each article  $a_j$  in Wikipedia is associated with the set of categories  $X_j$ . We represent the category  $x \in X_j$  as a real-valued vector  $\mathbf{m}_x \in \mathbb{R}^d$ .

*CBOW* model optimize following objective function:

$$\sum_{a_j \in D} \sum_{i=1}^{L_j} E(w_{j,i}, \frac{\sum_{w_c \in C_{j,i}} \mathbf{v}_{w_c} + \sum_{x \in X_j} \mathbf{m}_x}{|C_{j,i}| + |X_j|}). \quad (3)$$

*Skip-gram* model optimize following objective function:

$$\sum_{a_j \in D} \sum_{i=1}^{L_j} \sum_{w_c \in C_{j,i}} E(w_{j,i}, \mathbf{v}_{w_c} + \sum_{x \in X_j} \mathbf{m}_x). \quad (4)$$

Visualization of the CBOW is presented in Figure 3. Visualization of the Skip-gram is presented in Figure 4.

We tested with *CBOW* and *Skip-gram* architectures enriched with categories that are shown in Figures 3 and 4. The *CBOW* architecture is generally much faster and easier to train and gives a good performance. The *Skip-gram* architecture is training 10x slower and was unstable during our setup with categories.

## 5.1 Setup

This article extends [34] that deals with four different types of model architectures and how to incorporate the categories for training the word embeddings, in this work the Czech language has been chosen to test the model with the following setup: Model is initialized with categories that are uniformly distributed. During training the sentence from a training corpora, we add vectors of corresponding categories to actual context window. Motivation of our approach comes from Distributional hypothesis [10] that says: "words that occur in the same contexts tend to have similar meanings". If we are training with the categories, we believe they would behave with respect to the Distributional hypothesis.

## 6 Training

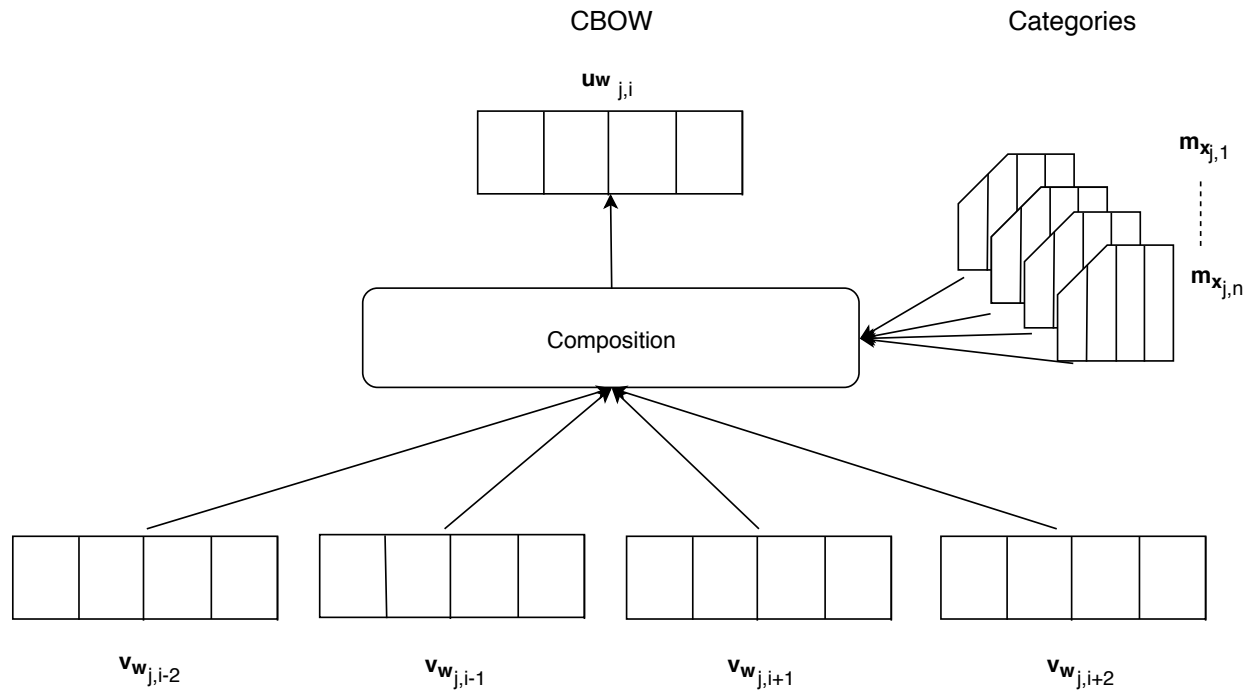
We previously tested our models on English Wikipedia dump from June 2016<sup>2</sup>. The XML dump consist of 5,164,793 articles and 1,759,101,849 words. We firstly removed XML tags and kept only articles marked with respective id, further we removed articles with less than 100 words or less than 10 sentences. We removed categories that has less than 10 occurrences in between all articles. We have removed the articles without categories. The final corpus used for training consist of 1,554,079 articles. Czech Wikipedia dump comes from March 2017. Detailed statistics on these corpora are shown in Tables 1 and 2. For an evaluation, we experiment with word analogy and a variety of word similarity datasets.

<sup>2</sup>[dumps.wikimedia.org](https://dumps.wikimedia.org)

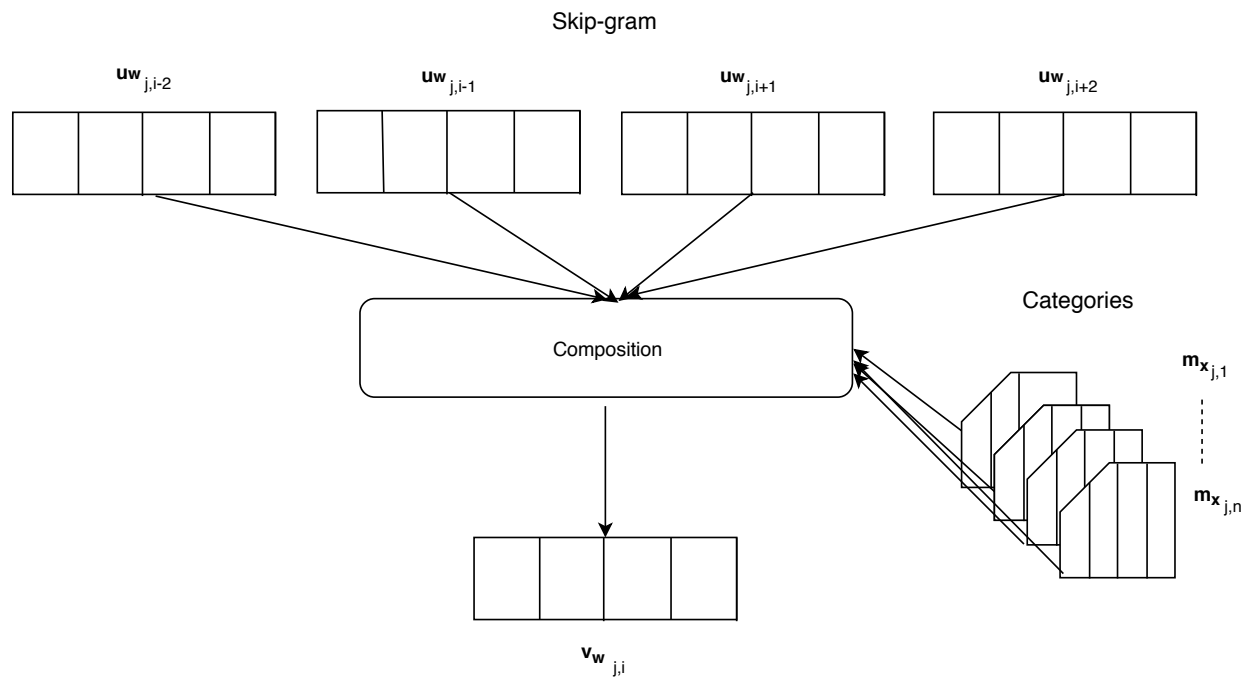
- **Word Similarity:** These datasets are conducted to measure the semantic similarity between pair of words. For English, these include WordSim-353 [6], RG-65 [28], RW [19], LexSim-999 [12], and MC-28 [23]. For Czech, only two datasets are available and these include RG-65 [15] and WordSim-353 [5]. Both datasets consists of translated word pairs from English and re-annotated with Czech native speakers.
- **Word Analogies:** Follow observation that the word representation can capture different aspects of meaning, [21] introduced evaluation scheme based on word analogies. Scheme consists of questions, e.g. which word is related to *man* in the same sense as *queen* is related to *king*? The correct answer should be *woman*. Such a question can be answered with a simple equation:  $\text{vec}(\textit{king}) - \text{vec}(\textit{queen}) = \text{vec}(\textit{man}) - \text{vec}(\textit{woman})$ . We evaluate on English and Czech word analogy datasets, proposed by [21] and [33], respectively. The word-phrases were excluded from original datasets, resulting in 8869 semantic and 10,675 syntactic questions for English (19,544 in total), and 6018 semantic. 14,820 syntactic questions for Czech (20,838 in total).

### 6.1 Training Setup

We tokenize the corpus data. We use simple tokeniser based on regular expressions. After model is trained, we keep the most frequent words in a vocabulary ( $|W| = 300,000$ ). Vector dimension for all our models is set to  $d = 300$ . We always run 10 training iterations. The window size is set 10 to the left and 10 to the right from the center word  $w_{j,i}$ , i.e.  $|C_{j,i}| = 20$ . The set of negative samples  $N$  is always sampled from unigram word distribution raised to 0.75 and has fixed size  $|N| = 10$ . We do not use the sub-sampling of frequent words. Process of parameter estimation process is described in detail in [9]. We prefixed categories to be unique in training and not interfering with words during training phase.



**Fig. 3.** Architecture of enriched CBOW model with categories



**Fig. 4.** Architecture of enriched Skip-gram model with categories

**Table 1.** Training corpora statistics. English Wikipedia dump from June 2016

	English (dump statistics)
Articles	5,164,793
Words	1,759,101,849
	English (final clean statistics)
Articles	1,554,079
Avg. words per article	437
Avg. number of categories per article	4.69
Category names vocabulary	4,015,918

**Table 2.** Training corpora statistics. Czech Wikipedia dump from June 2016

	Czech (dump statistics)
Articles	575,262
Words	88,745,854
	Czech (final clean statistics)
Articles	480,006
Avg. words per article	308
Avg. number of categories per article	4.19
Category names vocabulary	261,565

*fastText* is trained on our Wikipedia dumps (see results in Table 3 and 4). *LexVec* is tested only for English, trained on Wikipedia 2015 + NewsCrawl<sup>3</sup>, has 7B tokens, vocabulary of 368,999 words and vectors of 300d. Both (*fastText* and *LexVec*) models use character n-grams of length 3-6 as subwords. For a comparison with much larger training data (only available for English), we downloaded *GoogleNews100B*<sup>4</sup> model that is trained using Skipgram architecture on 100 billion words corpus and negative sampling, vocabulary size is 3,000,000 words.

## 7 Experimental Results and Discussion

We experiment with model defined in Section 5.1 and training Setup from Section 6.1.

As an evaluation measure for word similarity tasks we use Spearman correlation between system output and human annotations. For word

analogy task we evaluate by accuracy of correctly returned answers. Results for English Wikipedia are shown in Table 3 and for Czech in Table 4. These detailed results allow for a precise evaluation and understanding of the behaviour of the method. First, it appears that, as we expected, it is more accurate to predict entities when categories are incorporated.

### 7.1 Discussion

Distributional word vector models capture some aspect of word co-occurrence statistics of the words in a language [17]. Therefore, these extended models produce semantically coherent representations, if we allow to see shared categories between semantically similar textual data, the improvements presented in Tables 3 and 4 is the evidence of the distributional hypothesis.

Our model on English also outperforms *fastText* architecture [1] that is a recent improvement of Word2Vec with sub-word information. With our adaptation, the CBOW architecture give similar performance as the Skipgram architecture trained on much larger data. On RG-65 word similarity test and semantic oriented analogy questions in Table

<sup>3</sup><http://www.statmt.org/wmt14/translation-task.html>

<sup>4</sup><https://developer.syn.co.in/tutorial/bot/oscova/pretrained-vectors.html>

**Table 3.** Word similarity and word analogy results on English

	Model	Word similarity				Word analogy		
		WS-353	RG-65	MC-28	Simlex-999	Sem.	Syn.	Total
<b>Baselines</b>	fastText - SG 300d wiki	46.12	76.31	73.26	26.78	68.77	67.94	68.27
	fastText - cbow 300d wiki	44.64	73.64	69.67	38.77	69.32	81.42	76.58
	SG GoogleNews 300d 100B	68.49	76.00	80.00	46.54	78.16	76.49	77.08
	CBOW 300d wiki	57.94	68.69	71.70	33.17	73.63	67.55	69.98
	SG 300d wiki	64.73	78.27	82.12	33.68	83.64	66.87	73.57
	LexVec 7B	59.53	74.64	74.08	40.22	80.92	66.11	72.83
	CBOW 300d + Cat	63.20	78.16	78.11	40.32	77.31	68.68	72.13
	SG 300d + Cat	62.55	80.25	86.07	33.54	80.77	71.05	74.93

**Table 4.** Word similarity and word analogy results on Czech

	Model	Word similarity			Word analogy		
		WS-353	RG-65	MC-28	Sem.	Syn.	Total
<b>Baselines</b>	fasttext - SG 300d wiki	67.04	67.07	72.90	49.03	76.95	71.72
	fasttext - CBOW 300d wiki	40.46	58.35	57.17	21.17	85.24	73.23
	CBOW 300d wiki	55.9	41.14	49.73	22.05	52.56	44.33
	SG 300d wiki	65.93	68.09	71.03	48.62	54.92	53.74
	CBOW 300d + Cat	54.31	47.03	49.31	42.00	62.54	58.69
	SG 300d + Cat	62	57.55	64.64	47.03	54.07	52.75

3 it gives better performance. We can see, that our model is powerful in semantics.

There is also significant performance gain on WS-353 similarity dataset and English language. Czech generally perform poorer, because of less amount of data to train and also due to the fact of the language properties. The Czech has free word order and higher morphological complexity that influences the quality of resulting word embeddings, that is also the reason why the sub-word information tends to give much better results. However, our method shows significant improvement in Semantics, where the performance with Czech language has improved twofold (see Table 4).

The individual improvements of word analogy tests with CBOW architecture are available in Table 5. These detailed results allow for a precise evaluation and analyse the behaviour of our model. In Czech language, we see the biggest gain in understanding of category "Jobs". This semantic category is specific to Czech language as it distinguishes between feminine and masculine form of profession. However, we do not see much difference in section "Nationalities" that also

describes countries and masculine versus feminine form of its representatives. We think this might be caused of lack data from Wikipedia.

In Czech, we use mostly masculine form in articles when talking about people from different countries. In a section "Pronouns" that deals with analogy questions such as: "*I, we*" versus "*you, they*", we clearly cannot benefit from incorporating the categories. The biggest performance gain is as we expected in semantic oriented categories such as: *Antonyms, State-cities and Family-relations*. English gives slightly lower score in *Family-relations* section of analogy corpus.

However, as English semantic analogy questions are already hitting correlations above 80% and especially for this section already more than 90%, we believe that we are already hitting the maximal capabilities of machines and humans agreement. This is the reason why we bring up the comparison with highly inflected language. In [33] and [32] has been shown that there is a space for the performance improvement of current state-of-the-art word embedding models on languages from Slavic families. More information

**Table 5.** Detailed word analogy results**(a)** CZ with CBOW and Categories

Type	Baseline	Cat
<b>Antonyms (nouns)</b>	15.72	7.14
<b>Antonyms (adj.)</b>	19.84	46.20
<b>Antonyms (verbs)</b>	6.70	5.00
<b>State-cities</b>	35.80	50.57
<b>Family-relations</b>	31.82	50.64
<b>Nouns-plural</b>	69.44	75.93
<b>Jobs</b>	76.66	95.45
<b>Verb-past</b>	51.06	61.04
<b>Pronouns</b>	11.58	10.42
<b>Antonyms-acjectives</b>	71.43	81.82
<b>Nationalities</b>	20.40	21.31

**(b)** EN with CBOW and Categories

Type	Baseline	Cat.
<b>Capital-common-countries</b>	84.98	88.34
<b>Capital-world</b>	81.78	87.69
<b>Currency</b>	5.56	5.56
<b>City-in-state</b>	62.55	65.22
<b>Family-relations</b>	92.11	90.94
<b>Adjective-to-adverb</b>	25.38	29.38
<b>Opposite</b>	41.67	37.08
<b>Comparative</b>	79.14	78.82
<b>Superlative</b>	59.74	64.50
<b>Present-participle</b>	61.95	65.89
<b>Nationality-adjective</b>	91.39	98.69
<b>Past-tense</b>	63.66	66.59
<b>Plural</b>	74.19	71.67
<b>Plural-verbs</b>	62.33	46.33

about individual section of Czech word analogy corpus is described in [33].

With Czech language, we investigated a drop in performance of the Skip-gram model. This fact might be caused of not enough data for the reverse logic of training the Skip-gram architecture.

## 8 Conclusion

### 8.1 Contributions

Our model with global information extracted from Wikipedia significantly outperform the baseline CBOW model. It provide similar performance

compared with methods trained on much larger datasets.

We focused on currently widely used CBOW method and Czech language. As a source of global document (respective article) context we used Wikipedia that is available in 301 languages. Therefore, it can be adopted to any other language without necessity of manual data annotation. The model can help to the highly inflected languages such as Czech is, to create word embeddings that perform better with smaller corpora.

### 8.2 Future Work

We believe that using our method together with sub-word information can have even bigger impact on poorly resourced and highly inflected languages, such as Czech from Slavic family. Therefore, the future community work might lead to integrate our model to the latest architectures such as *fastText* or *LexVec* and improve the performance further from incorporating the sub-word information.

Also, we suggest to take a look into the other possibilities, how to extract useful information from Wikipedia and how to use it during training - such as references, notes, literature, external links, summary info (usually displayed on the right side of the screen) and others.

We provide the global information data and trained word vectors for research purposes<sup>5</sup>.

## Acknowledgements

This work was supported by the project LO1506 of the Czech Ministry of Education, Youth and Sports and by Grant No. SGS-2019-018 Data and Software Engineering for Advanced Applications. Computational resources were provided by the CESNET LM2015042 and the CERIT Scientific Cloud LM2015085, provided under the programme "Projects of Large Research, Development, and Innovations Infrastructures.

<sup>5</sup>[https://github.com/Svobik1/global\\_context/](https://github.com/Svobik1/global_context/)

## References

1. Bojanowski, P., Grave, E., Joulin, A., & Mikolov, T. (2017). Enriching word vectors with subword information. *Transactions of the Association for Computational Linguistics*, Vol. 5, pp. 135–146.
2. Brychcín, T. & Konopík, M. (2015). Latent semantics in language models. *Computer Speech & Language*, Vol. 33, No. 1, pp. 88–108.
3. Brychcín, T. & Svoboda, L. (2016). Uwb at semeval-2016 task 1: Semantic textual similarity using lexical, syntactic, and semantic information. *Proceedings of the 10th International Workshop on Semantic Evaluation (SemEval-2016)*, pp. 588–594.
4. Charles, W. G. (2000). Contextual correlates of meaning. *Applied Psycholinguistics*, Vol. 21, No. 4, pp. 505–524.
5. Cinková, S. (2016). Wordsim353 for czech. *International Conference on Text, Speech, and Dialogue*, Springer, pp. 190–197.
6. Finkelstein, L., Gabrilovich, E., Matias, Y., Rivlin, E., Solan, Z., Wolfman, G., & Ruppín, E. (2002). Placing search in context: The concept revisited. *ACM Transactions on information systems*, Vol. 20, No. 1, pp. 116–131.
7. Firth, J. R. (1957). A synopsis of linguistic theory, 1930-1955. *Studies in linguistic analysis*.
8. Gabrilovich, E. & Markovitch, S. (2009). Wikipedia-based semantic interpretation for natural language processing. *Journal of Artificial Intelligence Research*, Vol. 34, pp. 443–498.
9. Goldberg, Y. & Levy, O. (2014). word2vec explained: deriving mikolov et al.'s negative-sampling word-embedding method. *arXiv preprint arXiv:1402.3722*.
10. Harris, Z. S. (1954). Distributional structure. *Word*, Vol. 10, No. 2-3, pp. 146–162.
11. Hercig, T., Brychcín, T., Svoboda, L., Konkol, M., & Steinberger, J. (2016). Unsupervised methods to improve aspect-based sentiment analysis in czech. *Computación y Sistemas*, Vol. 20, No. 3, pp. 365–375.
12. Hill, F., Reichart, R., & Korhonen, A. (2015). Simlex-999: Evaluating semantic models with (genuine) similarity estimation. *Computational Linguistics*, Vol. 41, No. 4, pp. 665–695.
13. Huang, E. H., Socher, R., Manning, C. D., & Ng, A. Y. (2012). Improving word representations via global context and multiple word prototypes. *Proceedings of the 50th Annual Meeting of the Association for Computational Linguistics: Long Papers-Volume 1*, Association for Computational Linguistics, pp. 873–882.
14. Konkol, M., Brychcín, T., & Konopík, M. (2015). Latent semantics in named entity recognition. *Expert Systems with Applications*, Vol. 42, No. 7, pp. 3470–3479.
15. Krcmár, L., Konopík, M., & Jezek, K. (2011). Exploration of semantic spaces obtained from czech corpora. *DATESO*, pp. 97–107.
16. Landauer, T. K., Foltz, P. W., & Laham, D. (1998). An introduction to latent semantic analysis. *Discourse processes*, Vol. 25, No. 2-3, pp. 259–284.
17. Levy, O. & Goldberg, Y. (2014). Dependency-based word embeddings. *Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, volume 2, pp. 302–308.
18. Lund, K. & Burgess, C. (1996). Producing high-dimensional semantic spaces from lexical co-occurrence. *Behavior research methods, instruments, & computers*, Vol. 28, No. 2, pp. 203–208.
19. Luong, T., Socher, R., & Manning, C. (2013). Better word representations with recursive neural networks for morphology. *Proceedings of the Seventeenth Conference on Computational Natural Language Learning*, pp. 104–113.
20. McNamara, D. S. (2011). Computational methods to extract meaning from text and advance theories of human cognition. *Topics in Cognitive Science*, Vol. 3, No. 1, pp. 3–17.
21. Mikolov, T., Chen, K., Corrado, G., & Dean, J. (2013). Efficient estimation of word representations in vector space. *arXiv preprint arXiv:1301.3781*.
22. Mikolov, T., Sutskever, I., Chen, K., Corrado, G. S., & Dean, J. (2013). Distributed representations of words and phrases and their compositionality. *Advances in neural information processing systems*, pp. 3111–3119.
23. Miller, G. A. & Charles, W. G. (1991). Contextual correlates of semantic similarity. *Language and cognitive processes*, Vol. 6, No. 1, pp. 1–28.



24. **Pelletier, F. J. (1994).** The principle of semantic compositionality. *Topoi*, Vol. 13, No. 1, pp. 11–24.
25. **Pennington, J., Socher, R., & Manning, C. (2014).** Glove: Global vectors for word representation. *Proceedings of the 2014 conference on empirical methods in natural language processing (EMNLP)*, pp. 1532–1543.
26. **Riordan, B. & Jones, M. N. (2011).** Redundancy in perceptual and linguistic experience: Comparing feature-based and distributional models of semantic representation. *Topics in Cognitive Science*, Vol. 3, No. 2, pp. 303–345.
27. **Rohde, D. L., Gonnerman, L. M., & Plaut, D. C. (2004).** An improved method for deriving word meaning from lexical co-occurrence. *Cognitive Psychology*, Vol. 7, pp. 573–605.
28. **Rubenstein, H. & Goodenough, J. B. (1965).** Contextual correlates of synonymy. *Communications of the ACM*, Vol. 8, No. 10, pp. 627–633.
29. **Salle, A., Idiart, M., & Villavicencio, A. (2016).** Matrix factorization using window sampling and negative sampling for improved word representations. *arXiv preprint arXiv:1606.00819*.
30. **Salle, A. & Villavicencio, A. (2018).** Incorporating subword information into matrix factorization word embeddings. *arXiv preprint arXiv:1805.03710*.
31. **Shuai, X., Liu, X., Xia, T., Wu, Y., & Guo, C. (2014).** Comparing the pulses of categorical hot events in twitter and weibo. *Proceedings of the 25th ACM conference on Hypertext and social media*, ACM, pp. 126–135.
32. **Svoboda, L. & Beliga, S. (2018).** Evaluation of croatian word embeddings. *Eleventh International Conference on Language Resources and Evaluation (LREC 2018)*.
33. **Svoboda, L. & Brychcin, T. (2016).** New word analogy corpus for exploring embeddings of czech words. *International Conference on Intelligent Text Processing and Computational Linguistics*, Springer, pp. 103–114.
34. **Svoboda, L. & Brychcin, T. (2018).** Improving word meaning representations using wikipedia categories. *Neural Network World*, Vol. 523, pp. 534.
35. **Turney, P. D. & Pantel, P. (2010).** From frequency to meaning: Vector space models of semantics. *CoRR*, Vol. abs/1003.1141.
36. **Zanzotto, F. M., Korkontzelos, I., Fallucchi, F., & Manandhar, S. (2010).** Estimating linear models for compositional distributional semantics. *Proceedings of the 23rd International Conference on Computational Linguistics*, COLING '10, Association for Computational Linguistics, Stroudsburg, PA, USA, pp. 1263–1271.

Article received on 17/01/2019; accepted on 04/03/2019.  
Corresponding author is Lukáš Svoboda.



# Semi-Automatic Knowledge Graph Construction by Relation Pattern Extraction

Yingju Xia, Zhongguang Zheng, Yao Meng, Jun Sun

Fujitsu Research & Development Center Co. LTD., Beijing,  
China

yjxia@cn.fujitsu.com

**Abstract.** Knowledge graphs represent information in the form of entities and relation-ships between them. A knowledge graph consists of multi-relational data, having entities as nodes and relations as edges. The relation indicates a relationship between these two entities. Relation extraction is the key step to construct a knowledge graph. Conventional relation extraction methods usually need large scale labeled samples for each website. It's difficult to deal with the large number of relations and the various representations of each relation. This paper proposed a novel semi-automatic method that builds knowledge graph by extracting relation patterns and finding new relations. The proposed method models the relation pattern as a tag sequence and learns the pattern similarity metric using the existing relation instances. The pattern similarity is adopted to extract new patterns for existing relations. The new relations are detected by using the pattern similarity and clustering technique. The experimental results on large scale web pages show the effectiveness and efficiency of the proposed method.

**Keywords.** Semantic web, relation extraction, knowledge graph.

## 1 Introduction

Knowledge graphs model information in the form of entities and relationships between them [1]. This kind of relational knowledge representation has a long history in logic and artificial intelligence [2], for example, in semantic networks [3] and frames [4]. It has been used in the Semantic Web community with the purpose of creating a “web of data” that is readable by machines [5]. Knowledge graph is a powerful tool for supporting a large spectrum of search applications including ranking, recommendation, exploratory search, and web search [6, 7]. A knowledge graph aggregates

information around entities across multiple content sources and links these entities together.

There is a growing interest in automatically constructing knowledge graphs [8, 9, 10]. However, automatically constructing such graphs from noisy extractions presents numerous challenges [11, 12]. There are many literatures related to this topic. From the early information extraction [13, 14, 15] to special data extraction, e.g. the web table extraction [16, 17], and further, the relation extraction [18, 19, 20]. The methods range from rule based methods [21, 22] to supervised methods and semi-supervised methods [23-31].

In this study, we focus on extracting the special information from structured web and building a knowledge graph. A sample web page is shown in Fig. 1. We extract the structure information shown in the red rectangle and build a knowledge graph about the enterprises. Each page on this kind of websites contains one or more facts about a particular entity (defined as topic entity, which is the subject for all relations in this page). For example, the sample web page in Fig. 1 gives information such as “Date of Establishment”, “Head Office” and “Capitalization” about a company.

The company entity will be the subject of all the relations and can be omitted, in this case, the relations can be represented as (relation, object). Take Fig. 1 for example, there will be some relations such as (*Date of Establishment*, “February 6, 1936”), (*Representative Director*, “Yoshinori Yamashita”) and (*Capitalization*, “135.3 billion yen”).

However, building knowledge graph from webpages is not easy.

Company Data	
Company Name	Ricoh Company, Ltd.
Date of Establishment	February 6, 1936
Head Office	3-6, Nakamagome 1-chome, Ohta-ku, Tokyo 143-8555 Japan +81 3-3777-8111 > <a href="#">Map</a>
Representative Directors	<b>Yoshinori Yamashita</b> President and CEO > <a href="#">The board</a>
Capitalization	135.3 billion yen (as of March 31, 2018)
Consolidated Net Sales	2,063.3 billion yen (Year ended March 31, 2018)

**Fig. 1.** A sample web page about a company

However, building knowledge graph from webpages is not easy. There are two main problems:

1. There are always various representations for one relation. For example, the relation “Date of Establishment” on a company website may be presented as “*Date of Establishment*”, “*establishment date*”, “*establishment day*”, “*Date of Company Established*” and “*Found date*”. It is hard to find all the possible descriptions.
2. There are always various templates to generate relation (relation, object) among different websites thus makes the structure or layout, differ from website to website. Take the “Required Education” of the company jobs for example, the XPath On the website (www.careerbuilder.com) is: “/html/body/table/tbody/tr/td/table/tbody/tr[2]/td/table/tbody/tr[8]/td[1]”. While it is “/html/body/div[1]/div[2]/div[1]/div[4]/div[1]/div/dl/dt” on another website (www.monster.com).

Furthermore, the templates will change due to website revisions. Even in the webpages generated from the same template, the pages may differ due to the missing fields, varying numbers of instances and conditional formatting. All these problems make the relation extraction much difficult.

The conversional relation extraction methods learn extraction patterns from manual annotations [6, 24, 26, 27, 32, 33]. The manual annotation is an expensive and time-consuming step.

The CERES system [24] uses an entity-linking step in the annotation process to identify detailed

page topic entities, followed by a clustering process to identify the areas of pages corresponding to each relation.

This method can compete with annotation-based techniques in the literature. This paper presents a novel semi-automatic knowledge graph construction method with relation pattern extraction using similarity learning. The knowledge graph is building from structured web page. Each web page is presented as a DOM Tree [34], the sample (relation, object) is presented as tag sequence of the XPath. The vector of the (relation, object) pair is gotten from the embedding method and feed to the Siamese network to learn a similarity metric. The relation pattern is built from the seed instance and be continually optimized by iterative steps.

The knowledge graph can be built in a semi-automatic way. Given some instances of the relations for an entity, the system build the relation patterns and find more relation instances by the similarity metric. The new relation instances are also used to refine the existing relation patterns. The system can also find new relations by clustering method using the learned similarity metric. For example, to build a knowledge graph for enterprise, the system only need some instances for the existing relations (“Date of Establishment”, “Capitalization”, “address”, “website” and so on), the system builds pattern for each relation and extracts information from web pages. The system find more relation instances from web pages and refine the relation patterns.

By using the similarity metric learnt from relation instances and the clustering approach, the system

can find new relations such as “slogan” and build pattern for it.

The main contributions of this paper are listed as following:

1. A method is proposed that using tag sequence and its embedding to build the relation patterns.
2. The relation extraction pattern similarity is learnt from the tag sequence of seed instances by using a Siamese network. The relation patterns can achieve self-improvement by finding more and more instances using the similarity metric.
3. The proposed method can also be used to detect the new relations and build the extraction patterns for the new relations.

The rest of this paper is organized as follows. Section 2 presents the knowledge graph building method using pattern similarity based relation extraction. Section 3 shows the experimental results. Section 4 gives several conclusions and future works.

## 2 Method Introduction

There are several steps in our system:

1. Make the representation for the relation instances.
2. Learn the similarity between tag sequences.
3. Build extraction patterns from seed instances.
4. Refine the extraction patterns with new instances.

### 2.1 Representation of the Input Relation Instances

The inputs of this system are two relation instances from the webpages. The *relation* ( $R$ ) and *object* ( $O$ ) of each relation instance ( $R$ ,  $O$ ) will be embedded in a tag sequence of XPath like this:

$\langle \text{tag}_{R1} \rangle \langle \text{tag}_{R2} \rangle \dots \dots \langle \text{tag}_{Rm} \rangle \quad R \quad \langle \text{tag}_{O1} \rangle \langle \text{tag}_{O2} \rangle \dots \dots \langle \text{tag}_{On} \rangle \quad O$

This tag sequence will be used to present the relation instance.

Take the “*Capitalization*” relation as example, the relation instance is (*Capitalization*, *135.3 billion yen*) and the XPath for these two nodes are:

$\text{/html/body/table/tbody/tr/td/table/tbody/tr[2]/td/table/tbody/tr[8]/td[1]}$  *Capitalization*  
 $\text{/html/body/table/tbody/tr/td/table/tbody/tr[2]/td/table/tbody/tr[8]/td[2]}$  *135.3 billion yen*

This relation instance is represented as a tag sequence:

$( \langle \text{html} \rangle \langle \text{body} \rangle \langle \text{table} \rangle \langle \text{tbody} \rangle \langle \text{tr} \rangle \langle \text{td} \rangle \langle \text{table} \rangle \langle \text{tbody} \rangle \langle \text{tr}[2] \rangle \langle \text{td} \rangle \langle \text{table} \rangle \langle \text{tbody} \rangle \langle \text{tr}[8] \rangle \langle \text{td}[1] \rangle \text{Capitalization} \langle \text{html} \rangle \langle \text{body} \rangle \langle \text{table} \rangle \langle \text{tbody} \rangle \langle \text{tr} \rangle \langle \text{td} \rangle \langle \text{table} \rangle \langle \text{tbody} \rangle \langle \text{tr}[2] \rangle \langle \text{td} \rangle \langle \text{table} \rangle \langle \text{tbody} \rangle \langle \text{tr}[8] \rangle \langle \text{td}[2] \rangle \text{135.3 billion yen} )$

This tag sequence presents the layout information on the web page.

The idea of this paper is to learn the similarity between relation instances and build the relation pattern using the similarity. It is hard to give the similarity of the relation instances pair, but it is easy to give the label that whether these two relation instances belong to the same relation or not. In our system, we use ‘0’ to indicate the same relation and ‘1’ for different relations.

For example, if we have another relation instance (*capital fund*, *\$202.5 billion*) and the tag sequence:

$( \langle \text{html} \rangle \langle \text{body} \rangle \langle \text{div}[1] \rangle \langle \text{div}[2] \rangle \langle \text{div}[1] \rangle \langle \text{div}[4] \rangle \langle \text{div}[1] \rangle \langle \text{div} \rangle \langle \text{dl} \rangle \langle \text{dt}[6] \rangle \text{capital fund} \langle \text{html} \rangle \langle \text{body} \rangle \langle \text{div}[1] \rangle \langle \text{div}[2] \rangle \langle \text{div}[1] \rangle \langle \text{div}[4] \rangle \langle \text{div}[1] \rangle \langle \text{div} \rangle \langle \text{dl} \rangle \langle \text{dd}[7] \rangle \text{\$202.5 billion} )$

When we put these two tag sequences to the system, we also should give the label ‘0’ (same relation). It is a training instance for the system.

### 2.2 The Siamese Network for Sequence Similarity Calculation

The first step of this method is the tag sequence embedding. That is to make a vector for the input tag sequence so that similar tag sequences or tag sequence used in a similar context are close to each other in the vector space. In the free text analysis field, the word embedding is widely used, particularly in deep learning applications.

The word embeddings are a set of feature engineering techniques that transform sparse vector representations of words into a dense, continuous vector space, enabling system to identify similarities between words and phrases based on their context.

In this paper, we adopt the word embedding approach [35, 36] and trained a Skip-Gram word2vec model from the intermediate result of DOM tree parsing. In Fig. 2, the  $X_{11}$ ,  $X_{12}$ , ..., and  $X_{1n}$  are tags for the first tag sequence and the  $X_{21}$ ,  $X_{22}$ , ..., and  $X_{2n}$  for the second tag sequence. They will be convert into vectors through the embedding component. After that, these tag embeddings are combined into one vector as the embedding of the tag sequence. There are several ways to combine the tag embeddings. In this paper, we chose the concatenation operation due to experimental results. The concatenation operation is to concatenate the vectors of each tag one by one to make a big vector. Say, if we have 10 tag vectors and each vector has the dimension 128, then the concatenation vector will has the dimension 1280.

The vectors of the tag sequence are put into the feature map layers. The feature maps layer converts the tag sequence into a target space such that a simple distance in the target space approximates the "semantic" distance in the input space. Since the two feature maps layer share the same parameter  $W$ , this can be consider as the Siamese architecture [37, 38]. This network is suitable for recognition or verification applications where the number of categories is very large and not known during training.

Given a family of functions  $Gw(x)$  parameterized by  $W$ , the method seeks to find a  $W$  such that the similarity metric  $Ew(X_1, X_2) = \|Gw(X_1) - Gw(X_2)\|$  is small if  $X_1$  and  $X_2$  belong to the same extraction pattern, and large if they belong to different patterns. In our system, the structure of the feature map network is a 5 layers full-connected network. The output dimension of the feature map is set to 128.

In the training stage, let  $Y$  be a binary label of the pair,  $Y=0$  if the  $X_1$  and  $X_2$  belong to the same relation (genuine pair) and  $Y=1$  otherwise (impostor pair). Let  $W$  be the shared parameter vector that is subject to learning, and let  $Gw(X_1)$  and  $Gw(X_2)$  be the two points in the low-dimensional space that are generated by mapping

and  $X_1$  and  $X_2$ . Then our system use the  $Ew(X_1, X_2)$  to measures the similarity between  $X_1$  and  $X_2$ .

It is defined as  $Ew(X_1, X_2) = \|Gw(X_1) - Gw(X_2)\|$ . The loss function is of the form:

$$\ell(W) = \sum_{i=1}^p L(W, (Y, X_1, X_2)^i) \quad (1)$$

$$L(W, (Y, X_1, X_2)^i) = (1 - Y)L_G(Ew(X_1, X_2)^i) + YL_I(Ew(X_1, X_2)^i) \quad (2)$$

where  $(Y, X_1, X_2)^i$  is the  $i$ -th sample, which is composed of a pair of samples and a label,  $L_G$  is the partial loss function for a genuine pair,  $L_I$  the partial loss function for an impostor pair, and  $P$  the number of training samples.  $L_G$  and  $L_I$  should be designed in such a way that minimization of  $L$  will decrease the energy of genuine pairs and increase the energy of impostor pairs. A simple way to achieve that is to make  $L_G$  monotonically increasing, and  $L_I$  monotonically decreasing. In this paper, the  $L_G$  and  $L_I$  are:

$$L_G = \frac{2}{Q} (Ew)^3, \quad (3)$$

$$L_I = Qe^{-\frac{Ew}{Q}}. \quad (4)$$

Here the  $Q$  is a constant and is set to the upper bound of  $Ew$ . The  $Ew$  is the similarity metric which learned by the Siamese network, it is in the range  $[0, 1]$ .

### 2.3 Relation Patterns Building and Refining

Once we have learnt the similarity metric, we can use it to calculate the similarity of two input tag sequences. The tag sequence pair with similarity bigger than a given threshold can be used to build the same relation pattern. That means that, if we have some seed instances, we can use them and the similarity metric to find more similar instances. And build the relation patterns from these instances.

How to build the relation pattern using the instances? There are several ways to do this. In the rule scenario, we can deduce the regular expression from the detected relation instances and use it as the relation pattern. In this paper, we use the centroid point as the relation pattern.

**Table 1.** The Enterprise Knowledge Graph dataset

Column title	Column title
Name	43,680
Address	100,840
Person	24,297
homepage	17,651
phone number	20,783
rel-org	15,880
finance	15,218
size	9,462
date	14,872
product	23,590

The key technique of this system is the similarity learning method. With the similarity metric, we can collect more and more relation instances and then build better extraction pattern. Iteratively, the extraction pattern is used to find more relation instances.

The experimental results in Session 3 shows the performance of our similarity learning method. For example, in the “*Capitalization*” scenario, we have the instance:

```
(<html> <body> <table> <tbody> <tr> <td>
<table> <tbody> <tr[2]> <td> <table> <tbody>
<tr[8]> <td[1]> Capitalization <html> <body>
<table> <tbody> <tr> <td> <table> <tbody> <tr[2]>
<td> <table> <tbody> <tr[8]> <td[2]> 135.3
billion yen)
```

and

```
(<html> <body> <div[1]> <div[2]> <div[1]>
<div[4]> > <div[1]> <div> <dl> <dt[6] capital fund
<html> <body> <div[1]> <div[2]> <div[1]> <div[4]>
<div[1]> <div> <dl> <dd[7]> $202.5 billion ).
```

After some iterations, we find some new tag sequence belong to the “*Capitalization*” relation, say:

```
(<html> <body> <div[1]> <div[3]> <div[2]>
<div> <div[7]> <div[1]> capital amount <html>
<body> <div[1]> <div[3]> <div[2]> <div> <div[7]>
<div[2] US$65,000,000 ).
```

These new tag sequences are put into the collection of the “*Capitalization*” relation and used

to update the relation pattern. Then the updated relation pattern is used to collect new relation instances. This method can also be used to detect the patterns for new relations. For example, if we already have some relations about the job such as the “*Date of Establishment*”, “*Capitalization*”, “*address*” and “*website*”. The proposed system may get some relation instances clusters using clustering method. There may be cluster for a new relation, say, “*Number of Employees*”. Here are the relation instances for this new relation:

```
( <html> <body> <div[1]> <div> <div> <div[2]>
<div[2]> <div[1]> <div[1]> <dl[4]> <dt> Number of
Employees <html> <body> <div[1]> <div> <div>
<div[2]> <div[2]> <div[1]> <div[1]> <dl[4]>
<dd> 98,868).
```

```
(<html> <body> <div[2]> <div> <dl[10]> <dt>
Staff number <html> <body> <div[2]> <div>
<dl[10]> <dd> 1,678 ).
```

The relation name is different (“*Number of Employees*” and “*Staff number*”) and the format of the object is different also. They should be put into one cluster using the clustering method.

### 3 Experiments

#### 3.1 Data Set

We built a knowledge graph for enterprise using the proposed method. Firstly, we got a collection of websites about Japanese companies by search engine. Start from some manually labeled instances, we built a knowledge graph which contains 2,717,653 company entities and 22,257,276 triples. To show the performance of our method. We conducted a set of experiments, which use part of the data. We selected 10 relations to train the pattern similarity learning model. The instance number of each relation are shown in the Table 1.

#### 3.2 Experimental Results

The evaluation metric for similarity are:

- FA (False Accept Rate, the percentage of impostor pairs accepted),
- FR (False Reject Rate, the percentage of genuine pairs rejected),

- EER (Equal Error Rate, the point where FA equals FR).

To train and evaluate the similarity module, we need to build a set of tag sequence pairs, including genuine pairs and impostor pairs. There are 10 relations and total 286,276 instances. If we random select 1000 instances from each relation, we can build 9,999,000 genuine pairs and 90,000,000 impostor pairs.

Table 2 shows the experimental results of similarity learning, the system get EER 0.01 using the 10,000 instances (1,000 instances per relation). The iteration steps is about 500 to get the performance. The Fig. 3 shows the EER at different instance number. We can see that the more instances, the better performance. It trends to convergence when the instances number of each relation is about 1000.

The main scenario for the proposed method is to find new patterns and new relations. To evaluate the performance, we conduct experiments on unseen dataset. The unseen dataset means the test data are separated from the training data. In the enterprise knowledge graph case, we train the model on the some relations and test on the dataset with other relations.

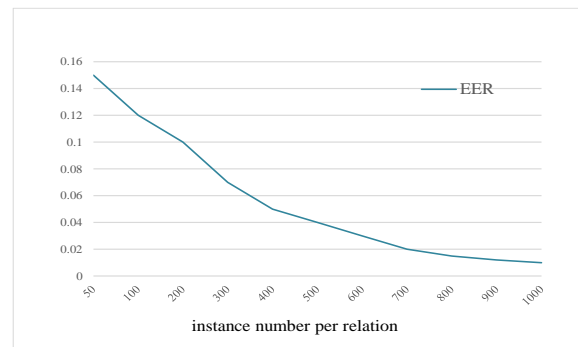
More concretely, we train the model on the previous 10 relations (name, address, person, homepage, phone number, rel-org, finance, size, date, and product). Then we test this model on the dataset with different relations (say, fax, email, Permission, Introduction, domain, office) shown on table 3. We find the optimal threshold on the valid dataset and use the threshold to get the FA and FR on test dataset. The EER is gotten from test dataset. The system got EER 0.05 on the unseen datasets. This enable the new patterns and relations find procedure.

In the scenarios that finding the new relations and getting the patterns, the clustering approach is adopted to put the similar instances into same group. To evaluate the new relation detection performance. We adopt the following steps and use the precision and recall metrics:

1. Input test data of  $k$  (for example,  $k=6$ ) categories.
2. Clustering, output  $k$  clusters.
3. Assign label for data in each cluster.

**Table 2.** Experimental Results of Similarity Learning

Instance number	Iteration steps	EER
10,000	100	0.04
10,000	500	0.01



**Fig. 3.** The impact of the instance number

**Table 3.** The unseen dataset

Relation	Number of instance
fax	5,911
email	3,586
permission	2,645
introduction	4,436
domains	13,347
offices	11,145

4. Compare the assigned labels with the true labels.
5. Using precision and recall (by Table 4) to evaluate the performance:

$$\text{Precision} = \text{TP} / (\text{TP} + \text{FP}),$$

$$\text{Recall} = \text{TP} / (\text{TP} + \text{FN}).$$

Two clustering methods (Hierarchy and K-Means) are evaluated. The experiments results shown in Table 5. From this table, we can see that K-Means got better results than hierarchy clustering method. It can be used to find the new relations.



**Table 4.** The metric for evaluated the clustering results

		Clustering labels	
		Y	N
True Label	P	True Positives (TP)	False Negatives (FN)
	N	False Positives (FP)	True Negatives (TN)

**Table 5.** The clustering experimental results

		Precision	Recall
Hierarchy Clustering	Fax	0.6976	0.775
	Email	0.7847	0.758
	Permission	0.7313	0.675
	Introduction	0.9973	0.366
	Domains	0.5903	1.00
	Office	0.9915	0.931
K-Means Clustering	Fax	0.7284	0.912
	Email	0.9134	0.77
	Permission	0.7414	0.671
	Introduction	0.9989	0.882
	Domains	0.8503	1.00
	Office	0.9883	0.93

## 4 Conclusion

This paper presents a new semi-automatic knowledge graph construction method by relation pattern extraction. The proposed method uses tag sequence to build the relation pattern.

A Siamese network is adopted to learn the similarity metric from the tag sequences. The proposed method builds the relation patterns and continually refines those patterns by finding more and more samples using the similarity metric.

It can also be used to detect patterns for the new relations. The future work includes finding new representation of the relation patterns and give the labels for new relations automatically.

## References

1. Nickel, M., Murphy, K., Tresp, V., & Gabrilovich, E. (2016). A review of relational machine learning for knowledge graphs. *Proceedings of the IEEE*,

- Vol. 104, No. 1, pp. 11–33. DOI: 10.1109/JPROC.2015.2483592.
2. **Davis, R., Shrobe, H., & Szolovits, P. (1993).** What is a knowledge representation?. *AI Magazine*, Vol. 14, No. 1, pp. 17–33. DOI: 10.1609/aimag.v14i1.1029.
  3. **Sowa, J.F. (2006).** Semantic networks. *Encyclopedia of Cognitive Science*. DOI: 10.1002/0470018860.s00065
  4. **Minsky, M. (1974).** A framework for representing knowledge. *MIT-AI Laboratory Memo 306*, pp. 1–81.
  5. **Berners-Lee, T., Hendler, J., & Lassila, O. (2001).** *The Semantic Web*. Scientific American, pp. 1–18.
  6. **Luna-Dong, X., Gabrilovich, E., Heitz, G., Horn, W., & Lao, N. (2014).** Knowledge vault: A web-scale approach to probabilistic knowledge fusion. *Proceedings of the 20th ACM (SIGKDD) International Conference on Knowledge Discovery and Data Mining. ACM'14*. pp. 601–610. DOI: 10.1145/2623330.2623623.
  7. **Voskarides, N., Meij, E., Tsagkias, M., de-Rijke, M., & Weerkamp, W. (2015).** Learning to explain entity relationships in knowledge graphs. *Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language*, pp. 564–574.
  8. **Fan, J., Kalyanpur, A., Gondeket, D.C., & Ferrucci, D.A. (2012).** Automatic knowledge extraction from documents. *Proceedings of the Joint Workshop on Automatic Knowledge Base Construction and Web-scale Knowledge Extraction*. Vol. 56, No. 3-4, pp. 1–10. DOI: 10.1147/JRD.2012.2186519.
  9. **Craven, M., DiPasquo, D., Freitag, D., McCallum, A., Mitchell, T., Nigam, K., & Slattery, S. (2000).** Learning to construct knowledge bases from the World Wide Web [J]. *Artificial intelligence*, Vol. 118, No. 1-2, pp. 69–113. DOI: 10.1016/S0004-3702(00)00004-7.
  10. **Weston, J., Bordes, A., Yakhnenko, O., & Usunier, N. (2013).** Connecting Language and Knowledge Bases with Embedding Models for Relation Extraction. *Processing, Conference on Empirical Methods in Natural Language*, pp. 1366–1371.
  11. **Jiang, S., Lowd, D., & Dou, D. (2012).** Learning to refine an automatically extracted knowledge base using markov logic. *IEEE 12th International Conference on Data Mining (ICDM)*, pp. 912–917. DOI: 10.1109/ICDM.2012.156.
  12. **Bordes, A., Glorot, X., Weston, J., & Bengio, Y. (2014).** A semantic matching energy function for learning with multi-relational data. *Machine Learning*, Vol. 94, No. 2, pp. 233–259.
  13. **Ferrara, E., De Meo, P., Fiumara, G., & Baumgartner, R. (2014).** *Web data extraction, applications and techniques: A survey Knowledge-based systems*. Vol. 70, pp. 301–323. DOI: 10.1016/j.knosys.2014.07.007.
  14. **Gottlob, G., Koch, C., & Pieris, A. (2017).** Logic, Languages, and Rules for Web Data Extraction and Reasoning over Data. *International Conference on Language and Automata Theory and Applications*. pp. 27–47. DOI: 10.1007/978-3-319-53733-7\_2.
  15. **Yao, X. & Van-Durme, B. (2014).** Information extraction over structured data: Question answering with freebase. *Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics*, pp. 946–966.
  16. **Lehmberg, O., Ritze, D., Meusel, R., & Bizer, C. (2016).** A large public corpus of web tables containing time and context metadata. *Proceedings of the 25th International Conference Companion on World Wide Web*. pp. 75–76. DOI: /10.1145/2872518.2889386.
  17. **Ritze, D., Lehmberg, O., Oulabi, Y., & Bizer, C. (2016).** Profiling the potential of web tables for augmenting cross-domain knowledge bases. *Proceedings of the 25th International Conference on World Wide Web, International World Wide Web Conferences Steering Committee*, pp. 251–261. DOI: 10.1145/2872427.2883017.
  18. **Song, M., Kim, W.C., Lee, D., Heo, G.E., & Kang, K.Y. (2015).** PKDE4J: Entity and relation extraction for public knowledge discovery. *Journal of Biomedical Informatics*, Vol. 57, pp. 320–332. DOI: 10.1016/j.jbi.2015.08.008.
  19. **Ji, G., Liu, K., He, S., & Zhao, J. (2017).** Distant Supervision for Relation Extraction with Sentence-Level Attention and Entity Descriptions. *AAAI*, pp. 3060–3066.
  20. **Heist, N. & Paulheim, H. (2017).** Language-agnostic relation extraction from wikipedia abstracts. *International Semantic Web Conference. Springer, Cham*, pp. 383–399. DOI: 10.1007/978-3-319-68288-4\_23.
  21. **Soderland, S. (1999).** Learning information extraction rules for semi-structured and free text. *Machine learning*, Vol. 34, No. 1-3, pp. 233–272. DOI: 10.1023/A:1007562322031.
  22. **Chang, C.H., Kaye, M., Girgis, M.R., & Shaalan, K.F. (2006).** A survey of web information extraction systems. *IEEE transactions on knowledge and data*

- engineering*, Vol. 18, No. 10, pp. 1411–1428. DOI: 10.1109/TKDE.2006.152.
23. **Mintz, M., Bills, S., Snow, R., & Jurafsky, D. (2009).** Distant supervision for relation extraction without labeled data. *Proceedings of the Joint Conference of the 47th Annual Meeting of the ACL and the 4th International Joint Conference on Natural Language Processing of the (AFNLP): Association for Computational Linguistics*, Vol. 2, pp. 1003–1011.
  24. **Lockard, C., Dong, X.L., Einolghozati, A., Shiralkaret, P. (2018).** CERES: Distantly Supervised Relation Extraction from the Semi-Structured Web. *Proceedings of the VLDB Endowment Endowment Homepage archive*, Vol. 11, No. 10, pp. 1084–1096. DOI: 10.14778/3231751.3231758.
  25. **Hoffmann, R., Zhang, C., Ling, X., Zettlemoyer, L., & Weld, D.S. (2011).** Knowledge-based weak supervision for information extraction of overlapping relations. *Proceeding HLT '11 Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies*, Vol. 1, pp. 541–550.
  26. **Hao, Q., Cai, R., Pang, Y. & Zhanget, L. (2011).** From one tree to a forest: a unified solution for structured web data extraction. *Proceedings of the 34th international ACM SIGIR conference on Research and development in Information Retrieval*. pp. 775–784. DOI: 10.1145/2009916.2010020.
  27. **Gentile, A.L., Zhang, Z., & Ciravegna, F. (2015).** Early steps towards web scale information extraction with lodie. *AI Magazine*, Vol. 36, No. 1, pp. 55–64. DOI: 10.1609/aimag.v36i1.2567.
  28. **Cohen, J.P., Ding, W., & Bagherjeiran, A. (2015).** Semi-supervised web wrapper repair via recur-sive tree matching. *CoRR, abs/1505.01303*, 2015.
  29. **Mironczuk, M.M. (2017).** The bigrams: the semi-supervised information extraction system from html: an improvement in the wrapper induction. *Knowledge and Information Systems*, Vol. 54, No. 3, pp. 711–776. DOI: 10.1007/s10115-017-1097-2.
  30. **Ortona, S., Orsi, G., Buoncristiano, M., & Furche, T. (2015).** Wadar: Joint wrapper and data re-pair. *PVLDB*, pp. 196–199.
  31. **Bronzi, M., Crescenzi, V., Merialdo, P., & Papotti, P. (2013).** Extraction and integration of partially overlapping web sources. *Proceedings of the VLDB Endowment VLDB Endowment Homepage archive*, Vol. 6, No. 10, pp. 805–816. DOI: 10.14778/2536206.2536209.
  32. **Kushmerick, N., Weld, D.S., & Doorenbos, R. (1997).** Wrapper induction for information extraction. *IJCAI*.
  33. **Gulhane, P., Madaan, A., Mehta, R.R., Ramamirtham, J., Rastogi, R., Satpal, S., Sengamedu, S.H., Tengli, A., & Tiwari, C. (2011).** Web-scale information extraction with Vertex. *The 27th International Conference on Data Engineering*, pp. 1209–1220. DOI: 10.1109/ICDE.2011.5767842.
  34. **Clark, J. & DeRose, S. (1999).** W3C. XML Path Language (XPath) Version 1.0, W3C Recommendation, pp. 1–34.
  35. **Levy, O. & Goldberg, Y. (2014).** Neural word embedding as implicit matrix factorization. *Advances in neural information processing systems*. pp. 1–9.
  36. **Mikolov, T., Sutskever, I., Chen, K., Corrado, G.S., & Dean, J. (2013).** Distributed representations of words and phrases and their compositionality. *Advances in neural information processing systems*, pp. 1–9.
  37. **Bromley, J., Guyon, I., LeCun, Y., Sicking, E., & Shah, R. (1994).** Signature verification using a "siamese" time delay neural network. *Advances in Neural Information Processing Systems*. pp. 737–744.
  38. **Chopra, S., Hadsell, R., & LeCun, Y. (2005).** Learning a similarity metric discriminatively, with application to face verification. *Computer Vision and Pattern Recognition*, Vol. 1, pp. 539–546.

Article received on 11/01/2019; accepted on 04/03/2019.  
Corresponding author is Yingju Xia.



# Identification of POS Tag for Khasi Language based on Hidden Markov Model POS Tagger

Sunita Warjri<sup>1</sup>, Partha Pakray<sup>2</sup>, Saralin Lyngdoh<sup>1</sup>, Arnab Kumar Maji<sup>1</sup>

<sup>1</sup> North-Eastern Hill University, Shillong, Meghalaya,  
India

<sup>2</sup> National Institute of Technology, Silchar, Assam,  
India

{sunitawarjri, parthapakray, saralylngdoh, arnab.maji}@gmail.com

**Abstract.** Computational Linguistic (CL) becomes an essential and important amenity in the present scenarios, as many different technologies are involved in making machines to understand human languages. Khasi is the language which is spoken in Meghalaya, India. Many Indian languages have been researched in different fields of Natural Language Processing (NLP), whereas Khasi lacks substantial research from the NLP perspectives. Therefore, in this paper, taking POS tagging as one of the key aspects of NLP, we present POS tagger based on Hidden Markov Model (HMM) for Khasi language. In this present preliminary stage of building NLP system for Khasi, with the analyses of the categories and structures of the words is started. Therefore, we have designed specific POS tagsets to categories Khasi words and vocabularies. Then, the POS system based on HMM is trained by using Khasi words which have been tagged manually using the designed tagsets. As ambiguity is one of the main challenges in POS tagging in Khasi, we anticipated difficulties in tagging. However, by running with the first few sets of data in the experimental data by using the HMM tagger we found out that the result yielded by this model is 76.70% of accurate.

**Keywords.** Natural language processing (NLP), computational linguistic, part of speech (POS), POS tagger, hidden Markov model (HMM).

## 1 Introduction

Natural Language Processing (NLP) deals with the inter-relation and inter-communication between the computer and natural human language by

combining the technology of artificial intelligent and computer science. The most important part of any NLP task is the issue of understanding the natural language. Application of NLP helps machines to learn, read and understand the human language, by simulating the human ability of understanding the language and by combining the technology of computational linguistics, computer science and artificial intelligence.

The most basic and important starting level of NLP for any language is POS tagging. POS in language processing is the aspect that deals with the identification of grammatical class of each word in a given sentence. POS is used in many fields of NLP such as Semantic Disambiguation, Phrase identification (chunking), Named Entity Recognition, Information Extraction, Parsing, etc. [2, 17]. The task of creating POS Tagger involves many stages such as: building tagsets, creating dictionary, considering the rules of the context and also checking the inflexions, dependent anomalies of the particular language.

Though, substantial work has already been carried out in different fields of NLP for Indian Language, Khasi lacks such study from the NLP perspective. Other Indian languages like Hindi, Bengali, Assamese, Manipuri, Marathi, Tamil, etc. have already been employed in Computational linguistic. In this paper we present POS tagger for Khasi language. Khasi Language is an official

language of the state Meghalaya, in North East India [16].

The name 'Khasi' classify both the tribe and as well as the language. Khasi Language is spoken in the Khasi Hills district of the state Meghalaya, India. Also, this language is spoken in the border area of Assam -Meghalaya as well as India-Bangladesh border. Khasi is a part of Mon-Khmer family which is branch of Austro-Asiatic, Southeast of Asia. There have been very limited research works in computational linguistics with regard to Khasi Language.

To perform research work on NLP there is a need of corpus in Khasi Language. Dataset or Corpus is basically a large and extensive collection of texts or words, which are used for analysis of any Natural Language. Corpus is an essential component for any Natural Language Processing research. Therefore, in this paper we describe tagger for Khasi Language based on supervised system HMM trained model.

The paper is organized as follows: Section 2 describes related works on POS Tagging; Section 3 describes Methodology used in HMM based POS Tagger system; Section 4 describes experimental results; Section 5 Conclusions and some future perspectives.

## 2 Literature Review

In this section, the existing relevant works in the POS tagging of different languages are presented. There are several existing research works on POS tagging on different languages such as Indian, English, German, Spanish, etc. whereby many researchers have also proposed different methods for POS tagging and show the achieved results.

A Hidden Markov Model (HMM) based Part of Speech (POS) Tagger for Hindi language as discussed in [5]. Indian Language (IL) POS tag set have been employed for the system. In the experimental result the HMM POS tagger acquires accuracy of 92%.

A POS tagging for Manipuri language demonstrated 69% of accuracy by using morphology driven POS tagger in [12]. This POS tagger uses 10917 unique words and three dictionaries

consisting of prefixes, suffixes and root words and also information with respect to the text content.

Part-of-speech (POS) Tagger for Malayalam Language using supervised learning based on Support Vector Machine (SVM) as discussed in [1]. For the tagger, tag-set consisting 29 tags was developed. The corpus with dataset consisting 180,000 words, the system achieved 94% of accuracy.

Part-of-Speech Tagging for Marathi Language discussed in [8] using Training data of 576 words with tag-set of 9 tags. Tokenization, Morphological analysis and Disambiguation process have been carried out for POS tagging. The author concluded with the attained accuracy of 78.82% by the system.

Kokborok language based on rule based, Conditional Random Field (CRF) and Support Vector Machines (SVM) for Part of Speech (POS) Tagger discussed in [9]. A tagged dataset of 42,537 words with 26 tag were used. The POS taggers methods attains the accuracies of 69% for rule based, 81.67% CRF based and 84.46% for SVM.

In [10] has discussed POS Tagging and Chunking using Conditional Random Fields and Transformation. The POS tagger accuracy achieved for CRF and TBL of about 77.37% for Telugu, 78.66% for Hindi, and 76.08% for Bengali and the chunker performance accuracy of 79.15% for Telugu, 80.97% for Hindi and 82.74% for Bengali respectively.

Part-of-Speech (POS) tagger for Bengali language in [3] has been reported. Tagging based on Hidden Markov Model (HMM) and Maximum Entropy (ME) stochastic taggers has been discussed. Study is conducted to improve the efficiency and performance of the tagger by using a morphological analyzer. An accuracy of 76.80% has been reported. The author reported to achieved good performance with the suffix information and morphological restriction on the grammatical categories for the supervised learning model.

In the paper [4] the POS tagger based on SVM and HMM for Bengali has been proposed. The author show the result as the accuracy of 79.6%

for the manually checked corpus consisting 0.128 million words.

The result of developed POS taggers were given as the accuracies of 85.56% for HMM, and 91.23% for SVM.

Part of Speech tagging for Assamese was reported in paper [11]. The system was design based on Hidden Markov Model approach (HMM). With tagsets consisting 172 tags and corpus consisting 10000 words which were manually tagged for training the system. The accuracy of 87% was achieved by the HMM POS tagger.

For khasi language the author in [13] had introduce tagsets consisting 61 tags. In the paper [14] the author had also introduce Morphological Analyzer for Khasi language. Using morphotactic rules the author had used dictionary consisting 8000 words. The analyzing system use based word of the word classes with grammatical relationship of subject verb object. For deriving the morphotactic rules the prefixes, infixes and suffixes of the words had been made used.

### 3 Methodology for HMM POS tagger

In this paper, the POS tagger for Khasi language based on Hidden Markov Model (HMM) as supervised learning has been employed. In the subsection below, we present the discussion about the methods that have been carried out in this work for building the supervised POS Tagger:

#### 3.1 Tag Sets

Tag is the label that is used to describe a grammatical class of information (these are: nouns, verbs, pronouns, prepositions, and so on). For example: NN could represent the Noun class, JJ the Adjective class, PRP the Pronoun class, etc. Each language has a different pattern, frequency, and speaking style. Thus, the grammatical class is also different for different languages. Therefore, for this work we have designed tagset consisting 54 tags for identifying the grammatical class or Part-of-Speech (POS) of Khasi Language [15] as listed in the Table 1 below.

**Table 1.** POS tagset for Khasi Language [15]

No.	Tag	Description
1	PPN	Proper nouns
2	CLN	Collective nouns
3	CMN	Common nouns
4	MTN	Material nouns
5	ABN	Abstract nouns
6	RFP	Reflexive Pronoun
7	EM	Emphatic Pronoun
8	RLP	Relative Pronouns
9	INP	Interrogative Pronouns
10	DMP	Demonstrative Pronouns
11	POP	Possessive Pronoun
12	CAV	Causative Verb
13	TRV	Transitive verb
14	ITV	Intransitive verb
15	DTV	Ditransitive verb
16	ADJ	Adjective
17	CMA	Comparative Adjective marker
18	SPA	Superlative Adjective marker
19	AD	Adverb
20	ADT	Adverb of Time
21	ADM	Adverb of Manner
22	ADP	Adverb of Place
23	ADF	Adverb of frequency
24	ADD	Adverb of degree
25	IN	Preposition
26	1PSG	1st Person singular common gender
27	1PPG	1st Person plural common gender
28	2PG	2ndPerson singular/plural common gender
29	2PF	2ndPerson singular/plural Feminine gender
30	2PM	2ndPerson singular/plural Masculine gender
31	3PSF	3rd Person singular Feminine gender
32	3PSM	3rd Person singular Masculine gender
33	3PPG	3rd Person plural common Gender
34	3PSG	3rd Person singular common Gender
35	VPT	Verb, present tense
36	VPP	Verb, present progressive participle
37	VST	Verb, past tense
38	VSP	Verb, past perfective participle
39	VFT	Verb, future tense
40	Mod	Modalities
41	Neg	Negation
42	CLF	Classifier

No.	Tag	Description
43	COC	Coordinating conjunction
44	SUC	Subordinating conjunction
45	CRC	Correlative conjunction
46	CN	Cardinal Number
47	ON	Ordinal number
48	QNT	Quantifiers
49	CO	Copula
50	InP	Infinitive Participle
51	PaV	Passive Voice
52	COM	Complementizer
53	FR	Foreign words
54	SYM	Symbols

For more details regarding the designed tag-sets of Khasi language can be found in [15] respectively.

### 3.2 Data Sets for Corpus Building

In linguistics, the corpus is the large number of data or written texts, which is collected for analyzing in computational linguistic. For this work, the corpus consists of the collected words and each word consists with its corresponding tags. It is found that no such Khasi corpus is available till date.

Therefore, in this work the Khasi corpus has been built. The corpus consists of Khasi language based on context, by collecting the written text from online Khasi newspaper. These raw texts are then tag appropriately to each word by using the designed tagset respectively. Tagging to words at this stage has been done manually. Therefore, we were able to create 7,500 words in our data set.

The corpus has been built painstakingly with care so that it can efficiently handle the problem of Ambiguity and also orthography. In this work, we aim to achieve standard Khasi corpus for POS tagging. Therefore, the dataset have also been built under the observation and validation of a linguistic expert from North Eastern Hill University, Department of Linguistic, Shillong, Meghalaya, India. Then, this dataset has been used for training and testing the tagger.

### 3.3 POS Identification based on HMM for Khasi Language

This subsection presents the steps for POS tagger based on Hidden Markov Model (HMM) that has been carried out in this work. Part-of-speech tagging is a sequence classification problem. The main objective of this supervised machine learning HMM trained model is to give the most i.e. the maximum probable tag  $y$  as outputs for the given word  $x$ , as shown in equation (1):

$$f(x) = \arg \max_{y \in Y} P(y|x). \quad (1)$$

The following are the elements consist in the HMM POS tagger system and the steps followed for calculating the efficiency of the tagger:

1. A finite set of words,  $W = \{w_1, w_2, \dots, w_n\}$ .
2. A finite tags,  $T = \{t_1, t_2, \dots, t_n\}$ .
3.  $n$  is the number in length.

Therefore, from our objective, to find the optimal tags sequence  $t^n$  we have equation (2) and (3):

$$t^n = \arg \max_{t^n} P(t^n|w^n), \quad (2)$$

$$t^n = \arg \max_{t^n} P(t^n)P(w^n|t^n). \quad (3)$$

NOTE: Prior probability -  $P(t^n)$  and Likelihood probability -  $P(w^n|t^n)$ .

4. Two properties of assumptions are considered in HMM based POS taggers, shown in equation (4) and (5):

i. The probability of bi-gram assumption:

$$P(t^n) = P(t_1)P(t_2|t_1)P(t_3|t_2)P(t_4|t_3) \dots P(t_n|t_{(n-1)}),$$

$$\approx \prod_{i=1}^n P(t_i|t_{i-1}). \quad (4)$$

ii. The assumption of Likelihood probability:

$$P(w^n|t^n) = P(w_1|t_1)P(w_2|t_2)P(w_3|t_3) \dots P(w_n|t_n),$$

$$\approx \prod_{i=1}^n P(w_i|t_i). \quad (5)$$



5. The POS Tagger use equation (6) for estimating the most probable sequence of tag.

$$(t^n) = \arg \max_{t^n} P(t^n | w^n):$$

$$\approx \arg \max_{t^n} \prod_{i=1}^n P(t_i | t_{i-1}) P(w_i | t_i). \quad (6)$$

6. Tag Transition probabilities  $P(t_i | t_{i-1})$  defining the probability of going from tag  $t_{i-1}$  to tag  $t_i$ , as shown in equation (7).

$$P(t_i | t_{i-1}) = \frac{\text{Count}(t_{i-1}, t_i)}{\text{Count}(t_{i-1})} \quad (7)$$

7. Word Emission probabilities  $P(w_i | t_i)$  defining the probability of emitting word  $w_i$  in tag  $t_i$  shown in equation (8):

$$P(w_i | t_i) = \frac{\text{Count}(t_i, w_i)}{\text{Count}(t_i)}. \quad (8)$$

For more details on supervised learning system based on HMM POS tagger and its assumption considered can be found in [7] respectively.

## 4 Experimental Results

In the subsection below, we present brief discussion on the experimental work conducted based on the corpus, with brief discussion on the result achieved and its analysis.

### 4.1 Corpus

As discussed in the methodology, the corpus has been manually designed and checked by the linguistic expert. In this work dataset or corpus of around 7,500 words has been used for training and 312 words for testing the HMM based POS Tagger. The corpus of Khasi language for this research has been collected from the online Khasi newspaper from [6], and the data collected comprise of the political news and article news had been used in the corpus. Some sample snap of the dataset that are manually tagged using the respective tagset is shown in Table 2. The right hand side represents the words and the left hand side represents its corresponding tags.

**Table 2.** Manually tagged dataset for training

Tag	Khasi Words
3PSF	Ka
CMN	Kynhun
COM	ba
VST	la
TRV	phah
EM	da
3PSM	u
CMN	Myntri
ADJ	Rangbah
3PSF	ka
PPN	Punjab
3PSM	u
FR	Capt.
PPN	Amanrinder
PPN	Singh
IN	hapoh
3PSF	ka
ABN	jingālam
POP	jong
3PSM	u
CMN	Myntri
3PSF	ka
CMN	tnad
FR	Water

### 4.2 Discussion on Some Challenges

During the collection of data and creating the dataset, we encountered some challenges; two main challenges in building the dataset for Khasi discussed in this paper are orthography and ambiguity. In '**Orthography**', the major problem is the spelling consistency. Spellings in Khasi have not been fully standardized. Different authors spell differently for the same words and that many words that are spelt alike have different meanings and are pronounced differently in different context. Whereas ambiguities are found in categorizing words that are spelt and pronounced alike but differ in categories when they are used in a sentence.

Assigning labels or tags to each word of a given sentence is a difficult task, because there are words that represent more than one grammatical part of speech. The challenge of POS tagging is the '**Ambiguity**'.

Some other structural are also there, but are kept out of this paper as they are not within the scope of the paper.

Keeping in mind these problems, we tried to consider by checking and accounting these problems wisely and built the dataset accordingly. The assumption that is considered for the word categories is based on the root category and prefix information. Therefore accordingly, we have designed the data sets to account these problems. Below are some of the examples cited, based on the challenges which are mentioned above:

**For example:** Orthography problem is as follows.

It is found that the orthography is very complex problem in Khasi language, as in some context words are different compare with the other; like the word *ia* in some context it is written as *ĩa*. There are many such words that are spell and written different by different people, some more of those words are like:

*ĩadei, ia dei, Khamtam, Kham tam, eiei, ei ei, ĩatreilang, ia trei lang, watla, wat la*, and so on.

**For example:** Ambiguity problem is as follows.

The Table 3 below show some of the ambiguous words of Khasi language:

**Table 3.** Some of the Khasi ambiguous words

Khasi word	Meaning	POS class
<i>Kot</i>	book	noun
<i>Kot</i>	reach	verb
<i>Kam</i>	work	noun
<i>Kam</i>	pace	verb
<i>lum</i>	hill	noun
<i>lum</i>	collect	verb
<i>bah</i>	sir	noun
<i>bah</i>	enough	adjective
<i>mar</i>	as soon as	adverb
<i>mar</i>	material	noun
<i>tam</i>	pick	verb
<i>tam</i>	over	adjective
<i>kham</i>	more	adjective
<i>kham</i>	hold	verb

Therefore according to our need of the data we have been considered and solve the problem.

### 4.3 Result

Using this HMM POS tagger based on the supervised learning method for the Khasi language, with the corpus of 7,500 words the system yield 76.70% of accuracy as a performance. Due to the unavailability of dataset or corpus and we had to make our own dataset, the accuracy of the tagging can be improved further by creating more data in the corpus. The Table 4 below show the comparison result with the other language using HMM approach.

### 4.4 Result Analysis

A part from the correct tagged words, in the experimental result some errors have also been analyses. It is found that some words are tagged incorrectly with the tag that does not belong to the respective word. As we have tag manually the Khasi words with respect to the content of context, therefore some words are tagged wrongly by the system due to ambiguity problem.

**For example, the word:**

*bha* is tagged as ADJ or AD.

*Namar* is tagged as COC or SUC.

*Hapdeng* is tagged as IN or ADP.

In the result it is found that for the words: *ia* it is tagged 12 times as InP and it is tagged 2 times as IN, *bah* is tagged as CMN 3 times and 1 time as ADJ, *dang* is tagged 1 time as AD and 1 time as VPP. Due to the present of ambiguous words in the annotated corpus it reduces the result accuracy. Therefore, with more annotated data there is high chance that the system will improve the result.

## 5 Conclusion and Future Works

As very limited works have been done for Khasi language in NLP till date, and tagging from the semantic and technical problems cited above have not been discussed at all, this work culminated as paper that addressed these issues from a larger perspective of NLP. The problems and issues being

**Table 4.** Comparison with others HMM POS tagger

Sl. no.	Paper Title	Language Used	Approach	corpus training	Accuracy
1	HMM based pos tagger for Hindi	Hindi	Hidden Markov Model	24 tags 3,58,288 words	92%.
2	Development of Marathi Part of Speech Tagger Using Statistical Approach	Marathi	Unigram, Bigram, Trigram and HMM Methods	26 lexical tags 1, 95,647 words	Unigram 77.38%, Bigram 90.30%, Trigram 91.46% and HMM 93.82%
3	Automatic Part-of-Speech Tagging for Bengali: An Approach for Morphologically Rich Languages in a Poor Resource Scenario Approach	Bengali	Maximum Entropy (ME) and HMM Methods	45,000 words	76.8%
4	Web-based Bengali News Corpus for Lexicon Development and POS Tagging	Bengali	HMM and SVM Methods	0.128 million words	SVM 1.23% HMM 85.56%
5	Part of Speech Tagger for Assamese Text	Assamese	HMM Method	10,000 words	87%
6	Identification of POS Tag for Khasi Language based on Hidden Markov Model POS Tagger (Our proposed method)	Khasi	HMM Method	7,500 words	76.70%

raised in this paper does not solve all the problems encounter in the POS tagging of Khasi, therefore, there is a future scope of accounting the other problems in future research.

Therefore, we aim to improve the result of the system by introducing more data in the corpus, for both training and testing. We also aim to develop some syntactic rules for Khasi language and to employ them in POS tagger for good results. This will help us to evaluate good performance of the POS tagger of Khasi language.

## Acknowledgement

Authors would like to acknowledge the Centre for Natural Language Processing, National Institute of Technology Silchar, 788010, Assam, India for this work.

## References

1. **Antony, P., Mohan, S. P., & Soman, K. (2010).** SVM based part of speech tagger for Malayalam. *Recent Trends in Information, Telecommunication and Computing (ITC), 2010 International Conference on*, IEEE, pp. 339–341.
2. **Chowdhury, G. G. (2003).** Natural language processing. *Annual review of information science and technology*, Vol. 37, No. 1, pp. 51–89.
3. **Dandapat, S., Sarkar, S., & Basu, A. (2007).** Automatic part-of-speech tagging for Bengali: An approach for morphologically rich languages in a poor resource scenario. *Proceedings of the 45th Annual Meeting of the ACL on Interactive Poster and Demonstration Sessions*, Association for Computational Linguistics, pp. 221–224.
4. **Ekbali, A. & Bandyopadhyay, S. (2008).** Web-based Bengali news corpus for lexicon development and POS tagging. *Polibits*, Vol. 37, pp. 21–30.
5. **Joshi, N., Darbari, H., & Mathur, I. (2013).** HMM based POS tagger for Hindi. *Proceeding of 2013*

- International Conference on Artificial Intelligence, Soft Computing (AISC-2013)*, pp. 341–349.
6. **Mawphor (2017)**. Mawphor. [Online; accessed 07-Nov-2017].
  7. **Pakray, P., Majumder, G., & Pathak, A. (2018)**. An HMM based pos tagger for POS tagging of code-mixed Indian social media text. *Annual Convention of the Computer Society of India*, Springer, pp. 495–504.
  8. **Patil, H., Patil, A., & Pawar, B. (2014)**. Part-of-Speech tagger for Marathi language using limited training corpora. *IJCA Proceedings on National Conference on Recent Advances in Information Technology NCRAIT (4)*, Citeseer, pp. 33–37.
  9. **Patra, B. G., Debbarma, K., Das, D., & Bandyopadhyay, S. (2012)**. Part of Speech (POS) tagger for Kokborok. *Proceedings of COLING 2012: Posters*, pp. 923–932.
  10. **PVS, A. & Karthik, G. (2007)**. Part-of-speech tagging and chunking using conditional random fields and transformation based learning. *Shallow Parsing for South Asian Languages*, Vol. 21, pp. 21–24.
  11. **Saharia, N., Das, D., Sharma, U., & Kalita, J. (2009)**. Part of speech tagger for Assamese text. *Proceedings of the ACL-IJCNLP 2009 Conference Short Papers*, Association for Computational Linguistics, pp. 33–36.
  12. **Singh, T. D. & Bandyopadhyay, S. (2008)**. Morphology driven Manipuri POS tagger. *Proceedings of the IJCNLP-08 Workshop on NLP for less privileged languages*, pp. 91–97.
  13. **Tham, M. J. (2012)**. Design considerations for developing a parts-of-speech tagset for Khasi. *Emerging Trends and Applications in Computer Science (NCETACS), 2012 3rd National Conference on*, IEEE, pp. 277–280.
  14. **Tham, M. J. (2013)**. Preliminary investigation of a morphological analyzer and generator for Khasi. *Emerging Trends and Applications in Computer Science (ICETACS), 2013 1st International Conference on*, IEEE, pp. 256–259.
  15. **Warjri, S., Pakray, P., Lyngdoh, S., & Kumar Maji, A. (2018)**. Khasi language as dominant Part-of-Speech (POS) ascendant in nlp. *International Journal of Computational Intelligence & IoT*, Vol. 1, No. 1, pp. 109–115.
  16. **Wikipedia contributors (2018)**. Khasi language — Wikipedia, the free encyclopedia. [Online; accessed 02-Feb-2018].
  17. **Wilks, Y. & Stevenson, M. (1998)**. The grammar of sense: Using 6 part-of-speech tags as a first step in semantic disambiguation. *Natural Language Engineering*, Vol. 4, No. 2, pp. 135–143.

Article received on 30/01/2019; accepted on 20/03/2019.  
Corresponding author is Sunita Warjri.

# Extracting Context of Math Formulae Contained inside Scientific Documents

Amarnath Pathak<sup>1</sup>, Ranjita Das<sup>1</sup>, Partha Pakray<sup>2</sup>, Alexander Gelbukh<sup>3</sup>

<sup>1</sup> National Institute of Technology Mizoram,  
Department of Computer Science and Engineering,  
India

<sup>2</sup> National Institute of Technology Silchar,  
Department of Computer Science and Engineering,  
India

<sup>3</sup> Instituto Politécnico Nacional,  
Mexico

{amarnath.cse.jrf, rdas}@nitmz.ac.in, partha@cse.nits.ac.in, www.gelbukh.com

**Abstract.** A math formula present inside a scientific document is often preceded by its textual description, which is commonly referred to as the context of formula. Annotating context to the formula enriches its semantics, and consequently impacts the retrieval of mathematical contents from scientific documents. Also, with a considerable surety, a context can be assumed to be one of the Noun Phrases (NPs) of the sentence in which formula occurs. However, the presence of several different misleading NPs in the sentence necessitates extraction of an NP, which is more precise to the formula than the rest. Although a fair number of methods are developed for precise context extraction, it can be fascinating to prospect other competent techniques which can further their performances. To this end, this paper discusses implementation of an automated context extraction system, which follows certain heuristics in assigning weights to different candidate NPs, and tune those weights using a development set comprising annotated formulae. The implemented system significantly outperforms nearest noun and sentence-pattern based methods on the ground of F-score.

**Keywords.** Context extraction, math information retrieval, NTCIR, parser, noun phrase.

## 1 Introduction

Increased research in Science, Technology, Engineering and Mathematics (STEM) disciplines has boosted the count of scientific documents, which are majorly constituted of math formulae. As a consequence, a number of Math Information Retrieval (MIR) systems, which can retrieve mathematical contents alongside plain text, have come into being. Math tasks [1, 2, 21] of NII Testbeds and Community for Information access Research (NTCIR) conferences [7, 6, 8] have also triggered widespread development of competent MIR systems.

Current MIR systems are either adapted versions of conventional text-search engines [11] or the systems developed from scratch [14, 15, 17]. While the text-search engines adapted for MIR perform linearization and plain text matching to retrieve formulae, the MIR systems developed from scratch employ novel formula indexing and search techniques. Although the challenges in designing such math-aware systems are enormous, the inability to account for ambiguity

of formula, whereby the same formula may have different alternative interpretations, can cause severe performance degradation. Consider, for example, the ambiguous formula ( $c = \sqrt{a^2 + b^2}$ ), which may exhibit following different meanings in two different documents:

- (a) Using Pythagorean theorem to compute hypotenuse (c) of a right-angled triangle, whose base is 'b' and perpendicular is 'a'.
- (b) Computing linear eccentricity (c) of hyperbola, with 'a' being distance from the center to the vertex and 'b' being the half distance between the asymptotes.

Given the two above-mentioned documents and a user query ( $c = \sqrt{a^2 + b^2}$ ) intending to retrieve search results for "linear eccentricity", a math-aware search engine [11], which only considers formula matching and discards the underlying semantics of the formulae, will also retrieve the irrelevant document containing "Pythagorean theorem". This diminishes precision score, hence reduces retrieval performance of MIR systems. Therefore, it becomes essential to extract most appropriate context from the surrounding text and perform semantification of formula by associating it with the extracted context. Also, the semantification of formula eliminates the need for querying a formula using only formula. Instead, the end-users relish flexibility to specify a text query (say, "Kinetic Energy") for searching a formula (say,  $\frac{1}{2}mv^2$ ) inside document. Moreover, semantification facilitates improvement in comprehensibility of formula.

The work described in this paper is based on a reasonable assumption that the context of formula is one of the Noun Phrases (NPs) of the sentence containing formula (henceforth called target sentence). Therefore, the context extraction task reduces to parsing the target sentence, extracting all the candidate NPs, and devising an algorithm to select the most appropriate NP from among the diverse pool of candidate NPs. However, as the most appropriate NP does not adhere to a strict pattern, the task of context extraction turns out to be challenging.

The following three example situations elaborate on this particular insight.

#### Example 1.1:

*Often, momentum transfer is given in wavenumber units in reciprocal length  $Q = k_f - k_i$ .*

In example 1.1 above, the three candidate NPs for the context of formula ( $Q = k_f - k_i$ ) are: "momentum transfer", "wavenumber units" and "reciprocal length". Also, the most appropriate context is the NP ("momentum transfer") which occurs farthest from the formula.

#### Example 1.2:

*This is the simplest example of scattering of two colliding particles with initial momenta  $\vec{p}_{i1}, \vec{p}_{i2}$ .*

In example 1.2 above, the candidate NPs for the context of formula ( $\vec{p}_{i1}, \vec{p}_{i2}$ ) are: "the simplest example of scattering of two colliding particles with initial momenta", "the simplest example", "scattering of two colliding particles with initial momenta", "scattering", "two colliding particles with initial momenta", "two colliding particles" and "initial momenta". Out of all the seven candidate NPs, the most appropriate one (i.e. "initial momenta") occurs closest to the formula.

#### Example 1.3:

*Using this free-body diagram the torque required to lift or lower a load can be calculated:  $T_{raise} = \frac{Fd_m}{2} \left( \frac{l + \pi \mu d_m}{\pi d_m - \mu l} \right) = \frac{Fd_m}{2} \tan(\phi + \lambda)$ .*

In example 1.3 above, the candidate NPs for the context of formula are: "this free-body diagram", "the torque required to lift or lower a load", "the torque" and "a load". Also, the most appropriate context (i.e. "the torque required to lift or lower a load") is the longest of all NPs and appears somewhere in the middle of the parse tree generated for target line.

To summarize, the uncertainty in position of occurrence of the context in the target sentence poses challenge to the design of context extraction

system. Nevertheless, owing to the advantages of formula semantification, recent years have witnessed a surge in the research activities concerned with context extraction. Some such activities include use of nearest noun method [13, 10], sentence–pattern based method [10, 20] and machine learning approach [10, 20]. After having identified all the NPs in target sentence, the nearest noun method considers nearest NP to be the context of formula. The sentence–pattern based method works under the assumption that the formulae are often linked to their contexts through specific words or group of words, such as “denotes”, “describes”, “means”, “is given by” and so on. For instance, in example 1.3 above, the formula and context are linked through the pattern, namely “can be calculated”. However, the method incurs failure in retrieving context–formula pairs which do not adhere to such fixed patterns. The machine learning based method [10] pairs math formula with “all NP” and “minimal NP” in the target sentence and extracts features for each pair. Thereafter, each pair is fed to a binary classifier to decide if the NP is most appropriate context for the formula.

The main contribution of this paper lies in devising a context extraction system, which extracts target sentences from scientific documents, parses all such sentences using Stanford Shift-Reduce Constituency Parser<sup>1</sup>[22, 4], extracts all NPs from the parse trees of target sentences, assigns weight to different candidate NPs of a formula using certain heuristics, tunes the weights using a development set containing formulae and their respective gold contexts, and eventually extracts most appropriate contexts from target sentences of test formulae. The implemented system performs reasonably well in comparison to other competent systems.

Rest of the paper is structured as follows: Section 2 reviews past works related to extraction of context of formula and identifier definition. Section 3 comprehensively describes working of different constituents of the implemented system. Section 4 describes experimental setup used to develop and evaluate the system. Section 5 presents experimental results and in-depth

analysis of results to comprehend strengths and weaknesses of the implemented system. Section 6 concludes the paper and points directions for future research.

## 2 Related Works

In past, the works related to extraction of contexts of formulae and definitions of their constituent identifiers have been prominent. As the two categories of works closely resemble, the following subsections elaborate on past developments related to both the categories.

### 2.1 Extraction of Formula Context

It is a usual practice to compare performance of any context extraction system with the performances of nearest noun and sentence–pattern based methods, which were introduced in the previous section. Sentence–pattern based method often uses the seven distinct patterns, described in [10, 19] and shown in Table 1, for discovery of context.

The work described in [9] views context extraction as a binary classification problem, wherein the description candidates associated with formulae are classified as correct or incorrect. Performances of nearest noun, sentence–pattern and machine learning methods for context extraction are compared and analyzed. The model using “All NP” approach and all possible feature augmented to machine learning approach depicts better performance than the model using minimal NP approach.

Work described in [20] focuses on connecting mathematical mentions, namely names, definitions and explanations, with corresponding mathematical expressions contained inside Japanese scientific papers. A Support Vector Machine (SVM) trained using features, such as basic patterns and linguistic information, helps select correct description for an expression and outperforms conventional pattern based method.

The guideline to annotate mathematical expressions with their respective definitions is described in [10]. The annotated data is used to examine performance of proposed context–extraction

<sup>1</sup><https://nlp.stanford.edu/software/srparser.html>

**Table 1.** Patterns used in sentence–pattern based method. MATH: math formula; DEF: definition of math formula i.e. context; OTHERMATH: other math formula

Sl.No.	Patterns
1	... denoted (as   by) MATH DEF
2	(let   set) MATH (denote   denotes   be) DEF
3	DEF (is   are)? (denoted   defined   given) (as   by) MATH
4	MATH (denotes   denote   (stand   stands) for   mean   means) DEF
5	MATH (is   are) DEF
6	DEF (is   are) MATH
7	DEF (OTHERMATH)* MATH

method. The proposed machine learning method extracts a set of 10 features (such as distance of candidate NP from the formula, Parts Of Speech (POS) tags of the text surrounding candidate NP, and so on) for candidate NPs and compares them with gold context. Under the constraint of strict matching, the machine learning method significantly outperforms nearest noun and pattern matching based methods.

The MARACHNA system [12] exploits Natural Language Processing (NLP) methods for extracting information from mathematical texts. More specifically, the MARACHNA generates ontologies for mathematical information extracted from different sources, and later stores them in a Knowledge Base (KB). The KB also stores different keywords and texts associated with a formula.

Concept Description Formula (CDF) approach [16] prospects coreference relation, if any, between the formula and context. The claim is made that extracting keywords using CDF and associating them with formulae will ease the task of MIR. Text preprocessing, text matching, pattern generation and pattern matching constitute key steps of CDF approach. Experimented using Wikipedia articles, the system depicts competence in finding coreference relation between text and formula.

An approach [5] to disambiguate mathematical expressions computes similarity between the words extracted from surrounding text of formula and a collection of term clusters derived from Content Dictionaries of OpenMath [3]. Subsequently, the cluster which shares highest similarity with the

words in surrounding text is considered to be the most accurate textual interpretation of the formula.

## 2.2 Extraction of Identifier Definition

Similar to formula, the meaning of an identifier may differ across documents or even across the formulae inside same document. For instance, the symbol '*E*' in a formula may designate electric field or Young's modulus. As the extractions of formula context and identifier definition share same underlying concerns, this subsection reviews some of the past works related to identifier definition extraction.

The Mathematical Language Processing (MLP) project [13] computes probabilities of identifier–definition pairs using POS based distance and sentence positions. Identifier definitions are discovered using pattern–based and statistical approaches. While the pattern–based approach uses 6 static patterns, the statistical approach extracts candidate definitions and ranks them using a weighted sum. Concretely, the statistical approach of MLP is a five–step process: (i) Detecting formulae from the documents (ii) Extracting identifiers from the formula (iii) Finding identifiers in surrounding text (iv) Finding candidate phrases/tokens for identifiers, and (v) Ranking candidate phrases/tokens using weighted sum.

The recall measure for statistical approach is found to be greater than that of pattern–based approach. Moreover, the statistical approach is least affected by the change in sentence structure.



Semantification of identifiers in formula is further improved through discovery of namespaces [18]. Although the concept of namespaces primarily applies to software development, the idea is extended to mathematical identifiers. Using NLP techniques, namespaces are discovered from the surrounding text of a formula. In summary, the identifier-definition extraction using namespace approach is a four-step process: (i) Automatic discovery of namespaces (ii) Clustering of documents (iii) Building namespaces, and (iv) Building namespace hierarchy.

### 3 System Description

Key constituents and working principle of the system are explained in subsequent subsections.

#### 3.1 Corpus Description

We experimented with the system using open source Wikipedia corpus<sup>2</sup> of NTCIR-12 MathIR task [21]. Unlike the arXiv corpus, the Wikipedia corpus is intended for non-technical users. Each document in the corpus contains scientific text alongside the formulae encoded using Presentation MathML, Content MathML and  $\text{\LaTeX}$ .

#### 3.2 Preprocessor

Owing to the presence of redundant HTML tags and spaces in the target sentences extracted from Wikipedia documents, the preprocessing of sentences was felt necessary prior to parsing. The job of preprocessor, therefore, is to remove all such redundant tags, spaces, links to footnotes and references, and so on. Table 3 shows sample examples of the preprocessings done by preprocessor.

<sup>2</sup>[www.cs.rit.edu/~rlaz/NTCIR12.MathIR.WikiCorpus-v2.1.0.tar.bz2](http://www.cs.rit.edu/~rlaz/NTCIR12.MathIR.WikiCorpus-v2.1.0.tar.bz2)

#### 3.3 Stanford Shift-Reduce Constituency Parser

The Stanford Shift-Reduce Constituency parser maintains the sentence on queue and the parse tree on stack. A set of transitions, namely shift, unary reduce, binary reduce, finalize and idle, are applied on the current state of the parse tree, unless the queue gets empty and the stack contains complete parse tree. Further, details related to the parser can be seen here<sup>3</sup>. Some sample processed target sentences and their respective parse trees as generated by the parser are shown in Table 4.

#### 3.4 Noun Phrase Extractor

After parsing, the parsed target sentences are fed to Noun Phrase Extractor (NPE), which extracts all different NPs from all the parsed sentences. Some examples of the candidate NPs extracted by the system are shown in Table 2. Up to this stage, a total of 4,919 instances (i.e. formulae, their target sentences, their parsed target sentences and all the candidate NPs present in parsed target sentences) are generated from 500 Wikipedia documents. Next, out of all such 4,919 instances, a set of 100 instances is selected as development set, and another different set of 100 instances is selected as test set. A development set is required to tune weights assigned to different candidate NPs of a target sentence. Moreover, for each instance in development set and test set, a gold context is manually selected from the candidate NPs. While the purpose of selecting gold contexts for development set is tuning of weights, the purpose of selecting gold contexts for test set is testing efficacy of the system for predicting correct context. Table 2 shows some sample entries of the development set. It also shows different candidate NPs from which the gold context is selected.

#### 3.5 Weight Assigner

Weight Assigner (WA) uses certain heuristics in assigning weights to different NPs as extracted by the NPE. Specifically, the following heuristics govern weight assignment:

<sup>3</sup><https://nlp.stanford.edu/software/srparser.html>

Table 2. Sample entries of the development set

Target sentence	Formula	Candidate NPs	Gold context
The units of specific contact resistivity are typically therefore in	$\Omega.cm^2$	<ul style="list-style-type: none"> <li>• The units of specific contact resistivity</li> <li>• The units</li> <li>• specific contact resistivity</li> </ul>	<ul style="list-style-type: none"> <li>• The units of specific contact resistivity</li> </ul>
The level of interaction can be measured by the Gravity model of trade	$I_{i,j} = \frac{p_i p_j}{d_{i,j}^2}$	<ul style="list-style-type: none"> <li>• The level of interaction</li> <li>• The level</li> <li>• interaction</li> <li>• the Gravity model of trade</li> <li>• the Gravity model</li> <li>• trade</li> </ul>	<ul style="list-style-type: none"> <li>• the Gravity model of trade</li> </ul>
Often, momentum transfer is given in wavenumber units in reciprocal length	$Q = k_f - k_i$	<ul style="list-style-type: none"> <li>• momentum transfer</li> <li>• wavenumber units</li> <li>• reciprocal length</li> </ul>	<ul style="list-style-type: none"> <li>• momentum transfer</li> </ul>
Assuming infinite planes, the magnitude of the electric field E is	$E = -\frac{\Delta\Phi}{d}$	<ul style="list-style-type: none"> <li>• infinite planes</li> <li>• the magnitude of the electric field E</li> <li>• the magnitude</li> <li>• the electric field E</li> </ul>	<ul style="list-style-type: none"> <li>• the magnitude of the electric field E</li> </ul>
The total Hamiltonian of an atom in a magnetic field is	$H = H_O + V + M$	<ul style="list-style-type: none"> <li>• The total Hamiltonian of an atom in a magnetic field</li> <li>• The total Hamiltonian</li> <li>• an atom</li> <li>• a magnetic field</li> </ul>	<ul style="list-style-type: none"> <li>• The total Hamiltonian of an atom in a magnetic field</li> </ul>
The following formula approximates the Earth's gravity variation with altitude	$g_h = g_0 \left( \frac{r_e}{r_e + h} \right)^2$	<ul style="list-style-type: none"> <li>• The following formula</li> <li>• the Earth 's gravity variation with altitude</li> <li>• the Earth 's gravity variation</li> <li>• the Earth 's</li> <li>• altitude</li> </ul>	<ul style="list-style-type: none"> <li>• the Earth 's gravity variation with altitude</li> </ul>

**Table 3.** Sample examples of preprocessing done by preprocessor

Original target sentence	Processed target sentence
$\langle /dl \rangle$ so Gaussian measure is a Radon measure; is not translation-invariant, but does satisfy the relation $\langle dl \rangle \langle dd \rangle \langle /dd \rangle \langle dt \rangle$	so Gaussian measure is a Radon measure; is not translation-invariant, but does satisfy the relation
The magnetic diffusivity is defined as: $\langle sup \rangle 1 \langle /sup \rangle$	The magnetic diffusivity is defined as:
$\langle h3 id="equation" \rangle$ Equation $\langle /h3 \rangle$ The mathematical equation for Boyle's law is	The mathematical equation for Boyle's law is

**Table 4.** Target sentences and their respective parse trees

Target sentence	Parse tree
The RMSD of an estimator	(ROOT (NP (NP (DT The) (NN RMSD)) (PP (IN of) (NP (DT an) (NN estimator))))))
The level of interaction can be measured by the Gravity model of trade	(ROOT (S (NP (NP (DT The) (NN level)) (PP (IN of) (NP (NN interaction)))) (VP (MD can) (VP (VB be) (VP (VBN measured) (PP (IN by) (NP (NP (DT the) (NN Gravity) (NN model)) (PP (IN of) (NP (NN trade))))))))))
The units of specific contact resistivity are typically therefore in	(ROOT (S (NP (NP (DT The) (NNS units)) (PP (IN of) (NP (JJ specific) (NN contact) (NN resistivity)))) (VP (VBP are) (ADVP (RB typically)) (ADVP (RB therefore)) (X (IN in))))
The wave number k is the absolute of the wave vector	(ROOT (S (NP (DT The) (NN wave) (NN number) (NN k)) (VP (VBZ is) (NP (NP (DT the) (JJ absolute)) (PP (IN of) (NP (DT the) (NP (NN wave) (NN vector))))))))

- (a) The WA only considers isolated NP (an NP which neither subsumes any NP nor is subsumed by any NP), maximal NP (an NP which subsumes one or more NPs, but is not subsumed by any other NP) and nearest NP (an NP which is nearest to the formula and may or may not be isolated and/or maximal NP) for weight assignment, and the rest candidate NPs are discarded. The isolated or maximal NP, which occurs farthest from the formula, is assigned a weight  $W_{begin}$  ( $W_{begin} \in \mathbb{R}$ ). Weight assigned to a subsequent maximal or isolated NP differs from its antecedent by an addend value  $f$  ( $f \in \mathbb{R}$ ). More specifically, the weight assigned to second farthest will be  $W_{begin} + f$ , third farthest will be  $W_{begin} + 2f$ , and so on.
- (b) In some cases, the nearest NP itself is the most appropriate context for the formula. Therefore, an additional weight, equal to

$W_{nearest}$  ( $W_{nearest} \in \mathbb{R}$  and  $W_{nearest} \geq 0$ ), is added to the existing weight of nearest NP.

- (c) Yet another heuristic is based on the observation that in most cases, if the farthest NP in original set of candidate NPs is an isolated NP, and the second farthest NP is a maximal NP, then the isolated NP is often trivial and the maximal NP is most appropriate context for formula. Consider the example 3.1 shown in Figure 1 to elucidate this point. In this example, the farthest NP is “*The following formula*”, which is also an isolated and trivial NP. Furthermore, the second farthest NP is “*the Earth's gravity variation with altitude*”, which is also a maximal NP and the gold context for formula.

Therefore, under such situation, an additional weight equal to  $W_{second}$  ( $W_{second} \in \mathbb{R}$  and

$W_{second} \geq 0$ ) is added to existing weight of the maximal and second farthest NP.

Table 5 describes weights assigned to different candidate NPs, using above-mentioned heuristics, under different example situations.

After the weights are assigned to different candidate NPs of a target sentence, and the weights are tuned using Weight and Addend Tuner (see subsection 3.6), the one having a maximum weight is predicted as the context of corresponding formula.

### 3.6 Weight and Addend Tuner

As the initial values of three different weight measures ( $W_{begin}$ ,  $W_{nearest}$  and  $W_{second}$ ) and the addend  $f$  may not be optimal, these values need to be tuned to ensure maximum F-score and, hence, optimal context prediction ability. The values of weight measures and addend are tuned using Weight and Addend Tuner (WAT), which compares the system predicted contexts (for development set) against the gold contexts to tune the weights and addend in trial and error fashion. Eventually, the WAT discovers best configuration (best set of values for  $W_{begin}$ ,  $W_{nearest}$ ,  $W_{second}$  and  $f$ ), which exhibits optimal performance (in terms of F-score) in context prediction.

### 3.7 System Testing

The efficacy of best configuration selected by WAT is examined using a test set comprising 100 previously unseen test instances. System predicted contexts are compared with the gold contexts of test instances, and the F-score is computed.

Figure 3 shows working principle and different constituents of the implemented system.

## 4 Experimental Design

To develop and test the system, following experimental setups are employed:

- (a) As mentioned in previous section, the system predicted contexts are evaluated on the ground of F-score. However, to compute F-score (see Equation 3), precision and recall measures need to be computed beforehand. While Precision  $P$  (see Equation 1) gives a measure of the correct context predictions out of the total contexts predicted, Recall  $R$  (see Equation 2) gives a measure of correct context predictions out of the total gold contexts:

$$P = \frac{\text{No. correct context predictions}}{\text{No. predicted contexts}}, \quad (1)$$

$$R = \frac{\text{No. context predictions}}{\text{No. gold contexts}}, \quad (2)$$

$$\text{F-score} = \frac{2 \times P \times R}{P + R}, \quad (3)$$

- (b) Also, as defined in previous section, the weight  $W_{begin}$  and the addend  $f$  may attain any real value during tuning, whereas the weights  $W_{second}$  and  $W_{nearest}$  only attain either 0 or positive real values.

## 5 Results and Analysis

System results and their comprehensive analysis are presented in this section. Following points regarding the implemented system and the predicted contexts are worth noting:

(1) System attains a maximum development set F-score of 67% for the following values of weights and addend:  $W_{begin} = 0.6$ ,  $W_{second} = 0.4$ ,  $W_{nearest} = 0.1$  and  $f = -0.2$ . A negative value of addend  $f$  is indicative of the fact that the value of weight  $W_{begin}$  is decremented by 0.2, every time, on going from farthest maximal/isolated NP to the nearest maximal/isolated NP.

(2) F-scores of the implemented system for development set and test set are 67% and 65%, respectively. The system predicts contexts (either correct or incorrect) for all the instances

**Example 3.1:**

**Target sentence:** *The following formula approximates the Earth's gravity variation with altitude*

**Formula:**  $g_h = g_0 \left( \frac{r_e}{r_e + h} \right)^2$

**Candidate NPs:** •The following formula •the Earth 's gravity variation with altitude •the Earth 's gravity variation •the Earth 's altitude

**Gold context:** the Earth 's gravity variation with altitude

**Fig. 1.** Example to explain heuristic

in development and test sets. Some sample examples of gold contexts, predicted contexts and their associated weights are shown in Table 6. For some of the test target sentences, there are more than one probable gold contexts (see example 5 of Table 6). In all such cases, the predicted context is considered relevant, if it is any one of the gold contexts.

(3) To assess the comparative strengths, the performance of proposed system is compared with those of nearest NP and sentence-pattern based methods (see Table 1 of Related Works section for different sentence patterns). Figure 2 shows F-scores of the three methods for development and test sets, and the scores are indicative of the fact that the proposed method significantly outperforms the two conventional and naive methods in predicting most appropriate context for a given formula. While the nearest noun method assumes nearest NP to be the context of formula, the sentence-pattern based method assumes presence of certain patterns between context and formula. Such naïve assumptions are not always correct; hence the poor performance.

(4) Different number of candidate NPs are extracted by the implemented system for different target sentences in development and test sets.

The graph shown in Figure 4 shows statistical distribution of number of candidate NPs in the two sets, which can be interpreted as follows: “3 candidate NPs are extracted for 15 target sentences in development set and 32 target sentences in test set. Similarly, 2 candidate NPs are extracted for 7 target sentences in development set and 4 target sentences in test set.” The plot is indicative of the fact that the two sets vary in terms of the instances corresponding to different number of candidate NPs. To further ascertain correlation, if any, between development and test sets, Pearson correlation coefficient ( $r$ ) is computed between number of instances in two sets corresponding to different number of candidate NPs. The value of  $r$  equal to 0.387 (more close to 0 than 1) confirms that the two sets are almost uncorrelated. Also, even though the development and test sets are almost uncorrelated, the implemented system delivers comparable performance. This confirms that the performance of system is independent of the nature of target sentences and, hence, the system is neither overfit nor underfit.

### 5.1 Error Analysis

Even though the proposed approach works effectively for substantial variety of test instances,

**Table 5.** Weights assigned to different candidate NPs of development set by WA

Target sentence	Candidate NPs	NPs selected by WA for weight assignment	Weights assigned
The level of interaction can be measured by the Gravity model of trade	<ul style="list-style-type: none"> <li>• The level of interaction</li> <li>• The level</li> <li>• interaction</li> <li>• the Gravity model of trade</li> <li>• the Gravity model</li> <li>• trade</li> </ul>	<ul style="list-style-type: none"> <li>• The level of interaction</li> <li>• the Gravity model of trade</li> <li>• trade</li> </ul>	$W_{begin}$ $W_{begin} + f$ $W_{nearest}$
Often, momentum transfer is given in wavenumber units in reciprocal length	<ul style="list-style-type: none"> <li>• momentum transfer</li> <li>• wavenumber units</li> <li>• reciprocal length</li> </ul>	<ul style="list-style-type: none"> <li>• momentum transfer</li> <li>• wavenumber units</li> <li>• reciprocal length</li> </ul>	$W_{begin}$ $W_{begin} + f$ $W_{begin} + 2f + W_{nearest}$
Assuming infinite planes, the magnitude of the electric field E is	<ul style="list-style-type: none"> <li>• infinite planes</li> <li>• the magnitude of the electric field E</li> <li>• the magnitude</li> <li>• the electric field E</li> </ul>	<ul style="list-style-type: none"> <li>• infinite planes</li> <li>• the magnitude of the electric field E</li> <li>• the electric field E</li> </ul>	$W_{begin}$ $W_{begin} + f + W_{second}$ $W_{nearest}$
The total Hamiltonian of an atom in a magnetic field is	<ul style="list-style-type: none"> <li>• The total Hamiltonian of an atom in a magnetic field</li> <li>• The total Hamiltonian</li> <li>• an atom</li> <li>• a magnetic field</li> </ul>	<ul style="list-style-type: none"> <li>• The total Hamiltonian of an atom in a magnetic field</li> </ul>	$W_{begin}$ $W_{nearest}$
Rectangular hyperbolas have eccentricity	<ul style="list-style-type: none"> <li>• Rectangular hyperbolas</li> <li>• eccentricity</li> </ul>	<ul style="list-style-type: none"> <li>• Rectangular hyperbolas</li> <li>• eccentricity</li> </ul>	$W_{begin}$ $W_{begin} + f + W_{nearest}$
The Rydberg constant is seen to be equal to	<ul style="list-style-type: none"> <li>• The Rydberg constant</li> </ul>	<ul style="list-style-type: none"> <li>• The Rydberg constant</li> </ul>	$W_{begin} + W_{nearest}$

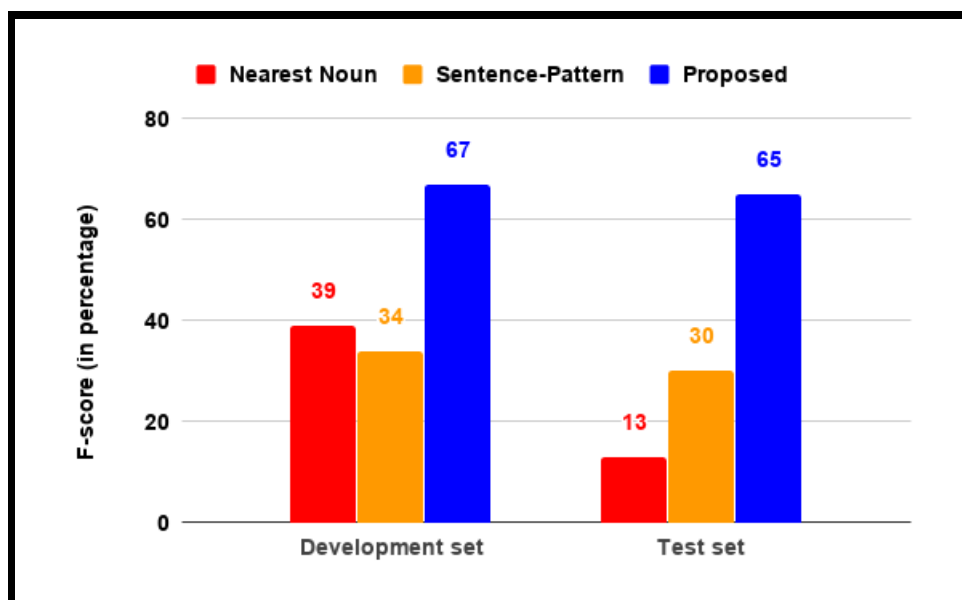


Fig. 2. Performance comparison of proposed method

the following shortcomings are worth considering:

- (a) The system incurs in failure in situations where the gold context is absent in the sentence containing formula. In example 5.1 (shown in Figure 5), the system incorrectly infers one of the NPs of target sentence to be the correct context, as the gold context (i.e. *“Rayleigh criterion”*) is present in some previous sentence and not in the sentence containing formula. Although it may be accounted by extending the window size from a single sentence to multiple sentences or even complete passage, such an attempt will lead to significant increase in number of NPs which are unrelated to the formula.
- (b) System lacks ability to combine different NPs in a systematic or meaningful way. More specifically, the system fails to account for situations wherein the gold context is intricate combination of two or more candidate NPs. For instance, in example 5.2 (shown in Figure 6), the gold context is combination of two candidate NPs, namely *“the pressure force”* and *“an isothermal fluid”*, but the system incorrectly predicts only partial gold context.

- (c) The system also incurs failure if the gold context does not adhere to the heuristics as mentioned in subsection 3.5. For instance, WA assigns weights only to maximal NP, isolated NP and nearest NP, and discards all other candidate NPs. However, in some cases, the gold context may be one of the minimal NPs, instead of being maximal NP, isolated NP or nearest NP. The gold context *“initial momenta”* is a minimal NP in example 5.3 (shown in Figure 7).

## 6 Conclusion and Future Directions

This paper proposes and implements a system, which can extract textual description (also called *“context”*) of math formula present inside scientific document. Preprocessor, Shift-Reduce Constituency Parser, Noun Phrase Extractor (NPE), Weight Assigner (WA), and Weight and Addend Tuner (WAT) form key constituents of the implemented system, which work in a sequential fashion to predict most appropriate context for a given formula. After the sentences containing formula (target sentences) are processed and parsed, different Noun Phrases (NPs) are extracted from

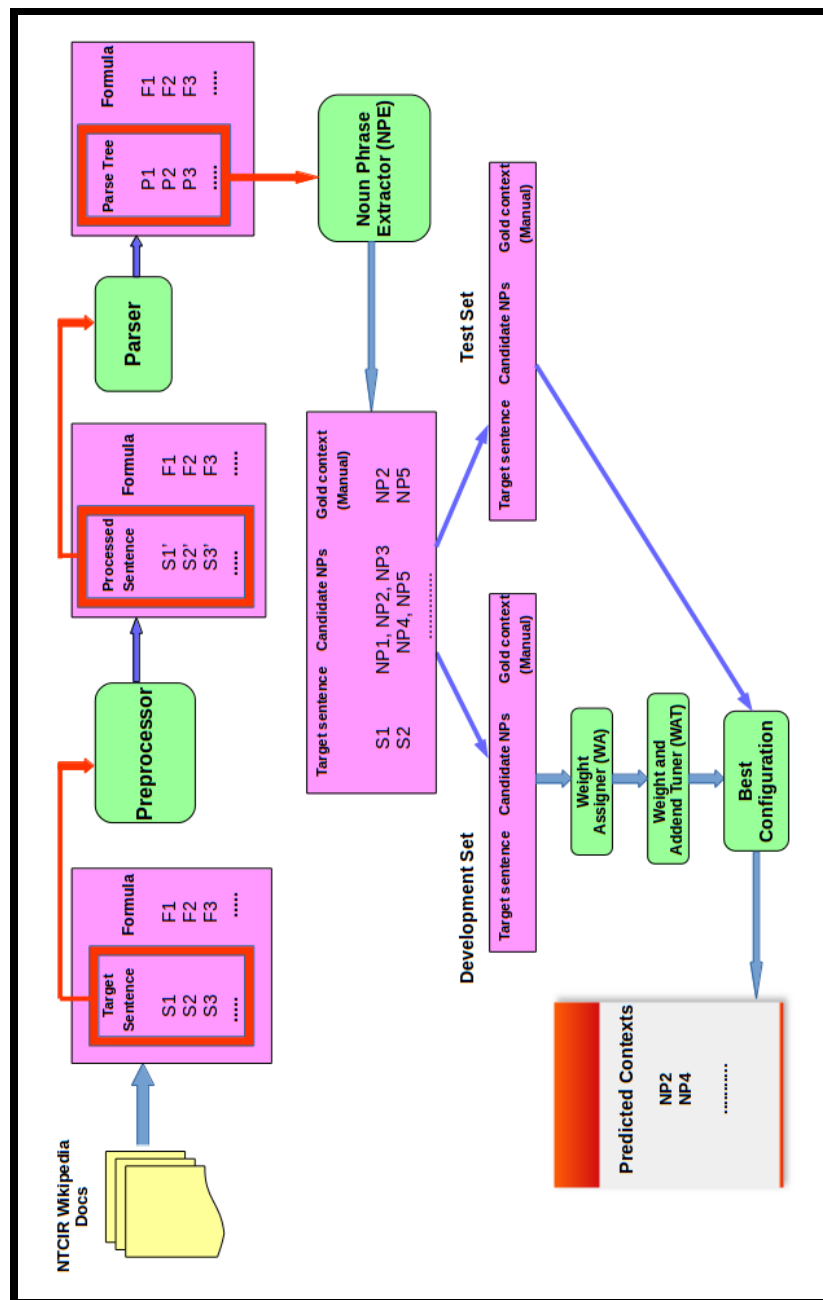


Fig. 3. Overall architecture of the implemented system



**Table 6.** Sample examples of gold contexts and predicted contexts

Target sentence	Gold context	Predicted context (weight)
Using this free-body diagram the torque required to lift or lower a load can be calculated:	the torque required to lift or lower a load	the torque required to lift or lower a load (0.8)
In its general form, the steering law can be expressed as	the steering law	the steering law (0.9)
The gravity depends only on the mass inside the sphere of radius	radius	the mass inside the sphere of radius (0.8)
The Young Equation relates the contact angle to interfacial energy	The Young Equation	The Young Equation (0.6)
The BEST theorem states that the number $ec(G)$ of Eulerian circuits in a connected Eulerian graph $G$ is given by the formula	(a) The BEST theorem (b) the number $ec(G)$ of Eulerian circuits in a connected Eulerian graph $G$	the number $ec(G)$ of Eulerian circuits in a connected Eulerian graph $G$ (0.8)

the parse trees using NPE. Eventually, the WA assigns weights to different NPs using certain heuristics, and the WAT tunes values of those weights using a development set containing gold contexts for target sentences.

Thereafter, the best set of weight values are used to predict context for test target sentences. The proposed method achieves a test set F-score of 65% and significantly outperforms the conventional nearest noun and sentence-pattern based methods of context extraction.

Followings are some of the future research directions worth exploring:

- (a) The WAT, as of now, uses trial and error to discover the best set of weight and addend values. Instead, in the future, multiple linear regression will be used to express F-score in terms of weights and addend. Subsequently, the constrained multivariable optimization, with constraints being  $W_{nearest} \geq 0$  and  $W_{second} \geq 0$ , will be used to discover the optimal values of weights and addend for which the F-score will be maximum.
- (b) Furthermore, it will be interesting to prospect the impacts of followings over performance of the system: (i) increase in development set size (ii) enabling support for judiciously

combining different candidate NPs, and (iii) extending the context window size from a single sentence to multiple sentences.

## Acknowledgement

The work presented here falls under the Research Project Grant No. YSS/2015/000988 and supported by the Department of Science & Technology (DST) and Science and Engineering Research Board (SERB), Govt. of India. The authors would like to acknowledge the Department of Computer Science & Engineering, National Institute of Technology Mizoram, India for providing infrastructural facilities and support. The fourth author acknowledges support of the SNI and the Instituto Politécnico Nacional via the grant SIP-IPN 20196437.

## References

1. Aizawa, A., Kohlhase, M., & Ounis, I. (2013). Ntcir-10 math pilot task overview. *Proceedings of the 10th NTCIR Conference*, Tokyo, Japan, pp. 654–661.
2. Aizawa, A., Kohlhase, M., Ounis, I., & Schubotz, M. (2014). Ntcir-11 math-2 task overview. *Proceedings of the 11th NTCIR Conference*, Tokyo, Japan, pp. 88–98.

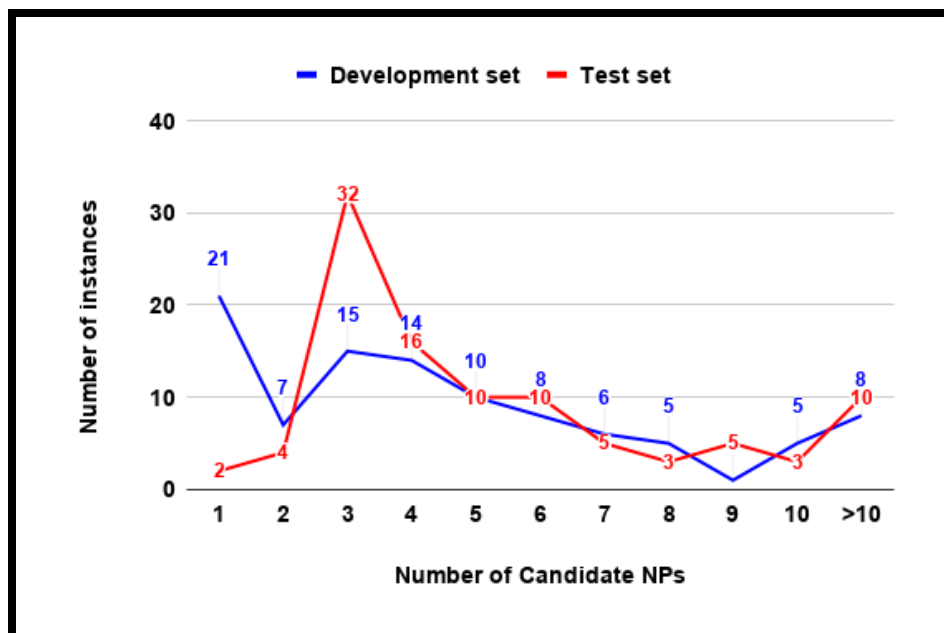


Fig. 4. Statistical distribution of number of candidate NPs in development and test sets

**Example 5.1:**

**Target sentence:** *If one considers diffraction through a circular aperture, this translates into:*

**Formula:**  $\theta = 1.220 \frac{\lambda}{D}$

**Gold context:** Rayleigh criterion

**Predicted context:** a circular aperture

Fig. 5. Error Analysis – I

3. Buswell, S., Caprotti, O., Carlisle, D. P., Dewar, M. C., Gaetano, M., & Kohlhase, M. (2004). The open math standard. Technical report, version 2.0. Technical report, The Open Math Society.
4. Goldberg, Y. & Nivre, J. (2012). A dynamic oracle for arc-eager dependency parsing. *Proceedings of COLING 2012*, pp. 959–976.
5. Grigore, M., Wolska, M., & Kohlhase, M. (2009). Towards context-based disambiguation of mathematical expressions. *The joint conference of ASCM*, Fukuoka, Japan, pp. 262–271.
6. Joho, H. & Kishida, K. (2014). Overview of ntcir-11. *Proceedings of the 11th NTCIR Conference on Evaluation of Information Access Technologies*, Tokyo, Japan, pp. 1–7.
7. Joho, H. & Sakai, T. (2014). Overview of ntcir-10. *Proceedings of the 10th NTCIR Conference*, Tokyo, Japan, pp. 1–7.

**Example 5.2:**

**Target sentence:** *For an isothermal fluid, the pressure force takes the form*

**Formula:**  $F_{fluid} = -k_B T_e \Delta n_e$

**Gold context:** the pressure force for an isothermal fluid

**Predicted context:** isothermal fluid

**Fig. 6.** Error Analysis – II**Example 5.3:**

**Target sentence:** *In the simplest example of scattering of two colliding particles with initial momenta of the form*

**Formula:**  $p_{i1}, p_{i2}$

**Candidate NPs:** •the simplest example of scattering of two colliding particles with initial momenta of the form •the simplest example •scattering of two colliding particles with initial momenta of the form •scattering •two colliding particles with initial momenta of the form •two colliding particles •initial momenta •the form

**Gold context:** initial momenta

**Predicted context:** the simplest example of scattering of two colliding particles with initial momenta of the form

**Fig. 7.** Error Analysis – III

8. Kishida, K. & Kato, M. P. (2016). Overview of ntcir-12. *Proceedings of the 12th NTCIR Conference on Evaluation of Information Access Technologies*, Tokyo, Japan, pp. 1–7.
9. Kristianto, G. Y., Aizawa, A., et al. (2014). Extracting textual descriptions of mathematical expressions in scientific papers. *D-Lib Magazine*, Vol. 20, No. 11, pp. 1–9.
10. Kristianto, G. Y., Nghiem, M.-Q., Matsubayashi, Y., & Aizawa, A. (2012). Extracting definitions of mathematical expressions in scientific papers. *Proceedings of the Annual Conference of JSAL*, Vol. JSAL2012, pp. 1–7.
11. Líška, M., Sojka, P., & Ružicka, M. (2013). Similarity search for mathematics: masaryk university team at the ntcir-10 math task. *Proceedings of the 10th NTCIR Conference on Evaluation of Information Access Technologies*, Tokyo, Japan, pp. 686–691.
12. Natho, N., Jeschke, S., Pfeiffer, O., & Wilke, M. (2008). Natural language processing methods for extracting information from mathematical texts. In *Advances in Communication Systems and Electrical Engineering*. Springer, pp. 297–308.
13. Pagael, R. & Schubotz, M. (2014). Mathematical language processing project. *arXiv preprint arXiv:1407.0167*.

14. Pathak, A., Pakray, P., & Gelbukh, A. (2018). A formula embedding approach to math information retrieval. *Computación y Sistemas*, Vol. 22, No. 3, pp. 819–833.
15. Pathak, A., Pakray, P., Sarkar, S., Das, D., & Gelbukh, A. (2017). Mathirs: Retrieval system for scientific documents. *Computación y Sistemas*, Vol. 21, No. 2, pp. 253–265.
16. Quoc, M. N., Yokoi, K., Matsubayashi, Y., & Aizawa, A. (2010). Mining coreference relations between formulas and text using wikipedia. *Proceedings of the Second Workshop on NLP Challenges in the Information Explosion Era (NLPPIX 2010)*, Beijing, China, pp. 69–74.
17. Ruzicka, M., Sojka, P., & Líska, M. (2016). Math indexer and searcher under the hood: Fine-tuning query expansion and unification strategies. *Proceedings of the 12th NTCIR Conference on Evaluation of Information Access Technologies*, Tokyo, Japan, pp. 331–337.
18. Schubotz, M., Grigorev, A., Leich, M., Cohl, H. S., Meuschke, N., Gipp, B., Youssef, A. S., & Markl, V. (2016). Semantification of identifiers in mathematics for better math information retrieval. *Proceedings of the 39th International ACM SIGIR conference on Research and Development in Information Retrieval*, ACM, Pisa, Italy, pp. 135–144.
19. Trzeciak, J. (1995). *Writing Mathematical Papers in English: A Practical Guide*. European Mathematical Society.
20. Yokoi, K., Nghiem, M.-Q., Matsubayashi, Y., & Aizawa, A. (2011). Contextual analysis of mathematical expressions for advanced mathematical search. *Polibits*, Vol. 43, pp. 81–86.
21. Zanibbi, R., Aizawa, A., Kohlhase, M., Ounis, I., Topic, G., & Davila, K. (2016). Ntcir-12 mathir task overview. *Proceedings of the 12th NTCIR Conference on Evaluation of Information Access Technologies*, Tokyo, Japan, pp. 299–308.
22. Zhu, M., Zhang, Y., Chen, W., Zhang, M., & Zhu, J. (2013). Fast and accurate shift-reduce constituent parsing. *Proceedings of the 51st Annual Meeting of the Association for Computational Linguistics*, volume 1, Sofia, Bulgaria, pp. 434–443.

Article received on 19/01/2019; accepted on 20/03/2019.  
Corresponding author is Partha Pakray.

# Identifying Repeated Sections within Documents

Girish K. Palshikar, Sachin Pawar, Rajiv Srivastava, Mahek Shah

TCS Research & Innovation, Pune,  
India

{gk.palshikar, sachin7.p, rajiv.srivastava, shah.mahek}@tcs.com

**Abstract.** Identifying sections containing a logically coherent text about a particular aspect is important for fine-grained IR, question-answering and information extraction. We propose a novel problem of identifying repeated sections, such as project details in resumes and different sports events in the transcript of a news broadcast. We focus on resumes and present four techniques (2 unsupervised, 2 supervised) for automatically identifying repeated project sections. The knowledge-based method is modeled after the human way closely. The other methods are based on integer linear programming and sequence labeling. The proposed techniques are general and can be used for identifying other kinds of repeated sections (and even non-repeating sections) in different types of documents. We compared the four methods on a dataset of resumes of IT professionals and also evaluated the benefits of identifying such repeated sections in practical IR tasks. To the best of our knowledge, this paper is the first to propose and solve the problem of repeated sections identification.

**Keywords.** Section identification, fine-grained IR, resume searching.

## 1 Introduction

Many semi-structured documents (e.g., resumes, news, medical reports, court judgments, financial analyst reports) are loosely organized in the form of *sections*, where each section contains logically coherent text about a particular aspect. E.g., a resume in the IT domain may contain sections related to Education, Employment History, Project Details, Trainings, Personal Information etc. Automatically identifying sections containing information of a specific type within particular types of documents is important for fine-grained

IR, question-answering and information extraction. E.g., if we are looking for people having hands-on experience in “SQL Server” in at least 2 projects, we should look only into the Project Details section (where the candidate lists all different projects she has worked on), and not, say, into the Trainings section of the resume.

In a semi-structured document, a section is often identified with a unique *section number*, and a *section title*. However, this identification is not uniform, even in the same type of documents. For example, in a resume, the section corresponding to past work experience has many different titles, such as Work History, Employment History, Work Experience, Experience Summary, Previous Employments etc. Thus identifying the section on past work experience in a resume can become difficult. In this paper, we focus on another kind of section identification: “identifying repeated sections”. In some type of documents, a same type of sections occurs multiple times. E.g., the resume of an IT person often has details on multiple projects that she has worked on, i.e., the resume contains multiple *project sections*, each of which gives the details of a particular project (see example in Table 1). All the project sections within a resume share common information elements, such as project title, client, duration, description, technology platforms used, role performed, team size etc. Different resumes may omit some of these information elements, and the order of these information elements may vary across resumes. These variations may even be present across project sections within the same resume.

As another example, legal contracts often contain many governance processes that describe

**Table 1.** An excerpt from an example resume where multiple repeated project sections are marked. Actual organization names in the resume are masked for anonymity

...	
PROJECTS	
SSSS FFFFF (Cargill, Brazil), May 2018 Present	Project1 begins
Internal application for client for its sales team.	Project1 continues
Application will be used to create and submit orders and view order history for its customers.	Project1 continues
Currently in planning phase. I am responsible as technical lead of team and currently understanding the requirements.	Project1 continues
WWW GGGG KKKKK (Belgium), Apr 2017 Apr 2018	Project2 begins
Hospital app for nurses visiting patients at home and keep track of medical records.	Project2 continues
Successfully delivered the project in requisite time and currently in UAT for final release.	Project2 continues
I was responsible for leading the offshore team on technical and design level and as supervisor.	Project2 continues
Also responsible for application development and issue resolution support.	Project2 continues
DDDD (DDDD JJJ), Jan 2017 Mar 2017	Project3 begins
A job hunting application for job seekers.	Project3 continues
Using the app, users can register themselves as potential job seekers.	Project3 continues
They can search and apply for jobs, check and update profile and keep a track of their job applications.	Project3 continues
I was responsible for design, requirement gathering and application development.	Project3 continues
...	

the sequences of actions to be taken when certain triggers occur. Table 2 shows examples of two governance processes in a construction contract<sup>1</sup>, which occur several pages apart and which are not explicitly marked as governance processes. Since contracts are complex and often hundreds of pages long, identifying such governance processes is critical for ensuring compliance to the contract. Treating each governance process as a “section”, we can apply the methods in this paper to automatically identify them. Similar repeating sections occur in many other types of documents such as court judgements, analyst reports, annual reports, and news.

While there is some research about identifying different types of sections in a document, we could not find any work that identifies *repeated* (sub-)sections of the same type. Hirohata et al. [4] proposed a CRF based approach to categorize sentences in scientific abstracts into 4 sections:

<sup>1</sup>[http://www.basnettdbr.com/pdfs/ConstCont\\_101117.pdf](http://www.basnettdbr.com/pdfs/ConstCont_101117.pdf)

objective, methods, results, and conclusions. Li et al. [8] is a supervised sequence labeling based approach for section classification which uses Hidden Markov Models. Shah et al. [10] proposes a CRF-based model for section identification, similar to ours. [12] propose a hierarchical information extraction framework to identify and label sections in a resume. [11] extracts different types of entities from resumes and uses them to improve ranking of resumes to match a job description; it does not deal with the problem of section identification. [2] is an unsupervised approach where they identify section labels using semantic relatedness (using word embeddings) between section title and contents and with predefined section labels. Guo et al. [3] proposed an unsupervised model which uses topic models to identify latent topics and their key linguistic features in input documents. Constraints are then induced from this information and sentences are mapped to their dominant section categories through a constrained unsupervised model.

**Table 2.** Example of governance processes in a construction contract (GP1: Governance Process 1, GP2: Governance Process 2)

...	
Contractor shall give written notice as necessary to Owner and/or Bank that Contractor's Work is completed and, if required, shall supply lien releases or receipts evidencing payment in full be filed relative to Contractor's Work.	GP1 begins
Owner and/or Bank shall have the right to make final inspection of Contractor's work within seven days after receipt of notice of completion and upon acceptance thereof by Owner and Bank, payment shall be made of the remaining balance due.	GP1 continues
...	
In the event that required work cannot be priced in advance of completion of such work, an Additional Work Authorization shall be executed.	GP2 begins
Such orders shall describe work to be completed, and shall specify method of calculating additional fees, materials, labor and services to be charged upon completion, and become part of this contract.	GP2 continues
Payment shall be due upon presentation of Contractor invoice.	GP2 continues
Additional time required shall be estimated and stated within the Additional Work Authorization.	GP2 continues
...	

Most of these previous approaches are dependent on dominant topics or lexical contents of the sections. As our work focuses on identifying repeated sections of the same type, our approaches also consider the structure of individual sections in the form of occurrence patterns of various section markers.

In this paper, we present four techniques for automatically identifying such repeated project sections in resumes: (i) knowledge-based, (ii) integer linear programming based (both *unsupervised*), (iii) sequence labeling using CRF, and (iv) sequence labeling using LSTM (both *supervised*).

## 2 Problem Definition

A given resume is a sequence of  $N$  sentences  $\langle 1, 2, \dots, N \rangle$ . The task is to identify all the project sections in this given resume, in terms of the start and end sentence numbers for each. To simplify the problem, we assume that the project sections are contiguous and non-overlapping. Then it is enough to identify the start sentence number for each project section. For example, if  $i$ -th project section starts on say line 68, then the  $(i - 1)$ -th project ends on sentence number 67. For the last project section, we use a simple heuristic rule to identify the end sentence number.

Let  $M$  denote the set of  $K$  markers. Each marker indicates the presence of a specific type of text. For the task of identifying project sections, we used  $K = 9$  markers: *blankline*, *project*, *projectnum*, *client*, *duration*, *role*, *teamsize*, *description*, *technology*.

Each of these markers detects the presence of some particular piece of information likely to be present in a project section. For example, the marker *project* corresponds to keywords indicating the presence of the title or name of project; e.g., Project, Module, Title, Name, Profile, Initiative.

We have defined a regular expression pattern to check whether or not a project marker is present in the given line. This regex not only detects the presence of required keywords but also performs additional checks; e.g., since *title* is also used for job designations, the regex checks for *absence* of designation indicating keywords (e.g., *manager*, *consultant*) near these keywords.

Thus we have  $K$  Boolean arrays each of length  $N$ ; for example, the array  $project[17] = 1$  implies that a project marker is present on sentence 17 of the given resume document. A single sentence may contain 0, 1 or more than one markers.

### 3 Repeated Sections Identification

#### 3.1 Knowledge-based Approach

A human HR executive can easily spot the project sections in any given resume. Given the endless variations in which the projects are written, this knowledge is quite non-trivial and does not consist of simple rules. In fact, we found that the humans have a dynamic, context-sensitive way of *rearranging* the project sections boundaries. We have tried to capture this expert knowledge in the form of an algorithm (Algorithm 1). The basic idea is that the human reader identifies the markers (already discussed), and then rearranges the project section starting point by understanding the spatial relationships among the marker positions.

The essence of this rearrangement is as follows. It is usually true that a project section begins with the *project* marker (e.g., project title), and all other markers (e.g., *client*, *teamsize* etc.) come afterwards within the project section. But occasionally some of the markers may occur just before the project title. Thus we need to recognize such deviations from the typical sequence of markers within a project section, and adjust the starting sentence of the project section accordingly. Since a lot of such variations among marker sequences occur in practical resumes, there is much heuristic post-processing still required after algorithm *identify\_project\_sections* (Algorithm 1), which we have omitted.

#### 3.2 ILP based Approach

We model the “repeated section identification” problem using the Integer Linear Programming (ILP) formulation. Table 3 depicts the input parameters, the constraints and the objective used for the ILP formulation. The 9 boolean arrays corresponding to project section markers described earlier are the input parameters. Another input for the ILP formulation is  $S$ , the maximum number of project sections possible for the current resume.

The **output representation** is in the form of two matrices ( $x$  and  $y$ ) of  $N \times S$  binary variables. If  $j^{th}$  project section begins at the  $k^{th}$  sentence, then the  $j^{th}$  column of  $x$  will contain all 0's before the

```

input :  $S = \{s_1, \dots, s_N\}$  sentences in given
        resume
input : project, projectnum, client, duration, role,
        teamsize, description, technology (Boolean
        arrays of length  $N$ )
output:  $P$  Boolean array of length  $N$ ,  $P[i] = 1$  if
        some project section starts at line  $i$ 
 $C :=$  array of  $N$  tuples, initialized with (0, NULL)
for  $i = 1$  to  $N$  do
    if project[ $i$ ]  $\wedge \exists$  another marker in  $i \pm K$  then
         $C[i] = (1, \text{project})$ 
for  $i = 1$  to  $N$  do
    if client[ $i$ ]  $\wedge C[j] \neq 1$  for  $i - K \leq j \leq i \wedge \exists$ 
        another marker in  $i \pm K$  then  $C[i] = (1, \text{client})$ 
for  $i = 1$  to  $N$  do
    if duration[ $i$ ]  $\wedge C[j] \neq 1$  for  $i - K \leq j \leq i \wedge \exists$ 
        another marker in  $i \pm K$  then
         $C[i] = (1, \text{duration})$ 
for  $i = 1$  to  $N$  do
    if  $C[i] == (1, \text{project})$  then
        Sequentially examine previous 10
        sentences starting at  $j = i - 1$  and stopping
        at any  $j$  where  $j$ -th sentence is either blank
        or does not contain any marker nor any ':';
        if  $j == i - 11 \wedge \text{isblank}(j)$  then  $P[j] = 1$ 
        else if  $j == i - 1 \wedge \exists i - 4 \leq k \leq i - 2$  s.t.
        project[ $k$ ]  $\wedge$  all sentences from  $k + 1$  to  $i - 1$ 
        are blank then  $P[k] = 1$ 
        else if  $j < i \wedge j \geq i - 10$  then  $P[j + 1] = 1$ 
        else  $P[i] = 1$ 
    else if
         $C[i] == (1, \text{client}) \vee C[i] == (1, \text{duration})$  then
        Sequentially examine previous 10
        sentences starting at  $j = i - 1$  and stopping
        at any  $j$  where both  $j, (j - 1)$ -th sentences
        are blank or  $j$ -th sentence contains any
        marker;
        if  $j < i \wedge j \geq i - 10$  then  $P[j + 1] = 1$ 
        else if  $j == i - 11 \wedge \exists j < k < i$  s.t.
        isblank( $k$ ) then  $P[k + 1] = 1$ 
        else if  $j == i - 11 \wedge \text{isblank}(j)$  then
         $P[j] = 1$ 
for  $i = 1$  to  $N$  do
    if  $P[i] == 1 \wedge$  this section marked due to
    project marker  $\wedge (i - 1)$ -th sentence has  $< 8$ 
    words  $\wedge \text{isblank}(i - 2)$  then
         $P[i - 1] = 1, P[i] = 0$ 

```

**Algorithm 1:** *identify\_project\_sections*



$k^{th}$  row and all 1's *after* that till the end. Similarly, if  $j^{th}$  project section ends at the  $k^{th}$  sentence, then the  $j^{th}$  column of  $y$  will contain all 0's before the  $k^{th}$  row and all 1's after that till the end. In other words,  $x$  is used to mark the beginning of individual project sections and  $y$  is used to mark the end. Hence,  $(x[i, j] - y[i, j])$  will be 1 if and only if  $i^{th}$  sentence is part of  $j^{th}$  section. Also, by definition of  $x$  and  $y$ ,  $(x[i, j] - y[i, j])$  will be 1 for consecutive sentences only. The **objective** is to minimize following 3 terms:

(1) Number of project markers which are not part of any section:  $\sum_{j=1}^S (x[i, j] - y[i, j])$  would be 0 only for sentences ( $i$ 's) which are not part of any section. Here, *duration* and *technology* markers are not considered as they also occur outside project sections (e.g., *duration* marker may be present in employment history).

(2) Number of project sections.  $x[N, j]$  is 1 if and only if  $j^{th}$  section is identified. Hence, the term  $\sum_{j=1}^S x[N, j]$  simply counts the number of project sections identified.

(3) Sentence numbers in which *project*, *projectnum* and *duration* markers occur in each section, relation to the sentence number corresponding to the beginning of that section. The term  $(x[i, j] - x[i - 1, j])$  will be 1 for one and only one  $i$  if  $j^{th}$  section is present, and hence the term  $\sum_{i=2}^N i \cdot (x[i, j] - x[i - 1, j])$  corresponds to the first sentence of the  $j^{th}$  section.

Various **constraints** are defined to capture different desired properties of the output project sections. Table 3 describes these constraints in detail. A separate ILP program is created for each resume and is solved to compute optimal feasible values for output variables, which in turn translate to predicted project sections.

### 3.3 Sequence Labeling Approaches

In addition to the unsupervised approaches (knowledge-based and ILP-based), we also explored two supervised approaches. Here, we model the "repeated section identification" problem as a sequence labeling task where an appropriate label is assigned to each element in a sequence. Each resume is represented as a sequence of sentences and 3 labels (B, I and O) are used

to represent the project section information. First sentence in each project section is labeled with B and all subsequent sentences within the section are labeled with I. Sentences which are not part of any project section are labeled with O. In order to learn a sequence labelling model, we explore following two approaches:

#### 3.3.1 Conditional Random Fields (CRF):

In this approach, each sentence is represented by a set of features which are designed to capture various characteristics of the sentence. Some of the representative features are as follows: presence of the project section markers in current, previous and next sentences, number of consecutive blank lines before and after each sentence, number of words in current sentence, distinct words present in current sentence. We use Conditional Random Fields (CRF) [6] for training a sequence labeling model, which is used to predict BIO labels for any unseen resume and predicted project sections can be derived from these labels.

#### 3.3.2 Long Short-Term Memory (LSTM):

The CRF-based approach requires explicit feature engineering for the task. Also, the markers-based features are dependent on accuracy and coverage of the regular expressions used to identify the markers. Hence, we also developed an LSTM-based [5] approach which bypasses the need for explicit feature engineering. Here, the model for section identification is built in two phases. Initially, our aim is to learn embedded representations for sentences in resumes. For this purpose, we train an LSTM-based sequence autoencoder [1] which consists of an LSTM encoder layer and another LSTM decoder layer. The detailed architecture is depicted in the Figure 1. Here, a sequence of words in a sentence is passed through the encoder LSTM layer, so that the output of the final time step provides the representation of the whole sentence. Words are represented using 100 dimensional pre-trained GloVe [9] word vectors. The decoder LSTM layer then tries to reconstruct the same sequence of the words using this representation. The model is trained in an unsupervised fashion to minimize

**Table 3.** ILP Formulation

<b>Input Parameters:</b>
$N$ : No. of sentences in the resume, $S$ : Max. no. of project sections possible, $project, client, projectnum, duration, role, teamsize$ and $description, technology$ : arrays of length $N$
<b>Decision Variables:</b>
$x$ : Binary matrix of size $N \times S$ , $x[i, j] = 1, \forall i \geq k$ s.t. $j^{th}$ section begins at the $k^{th}$ sentence $y$ : Binary matrix of size $N \times S$ , $y[i, j] = 1, \forall i \geq k$ s.t. $j^{th}$ section ends at the $k^{th}$ sentence
<b>Minimize:</b> $T_1 + T_2 + T_3$
$T_1 = \sum_{i=1}^N ((project[i] + client[i] + role[i] + teamsize[i] + description[i] + projectnum[i]) \cdot (1 - \sum_{j=1}^S (x[i, j] - y[i, j])));$ $T_2 = \sum_{j=1}^S x[N, j];$ $T_3 = 0.001 \cdot \sum_{j=1}^S (\sum_{i=1}^N i \cdot (project[i] + projectnum[i] + client[i] + duration[i]) \cdot (x[i, j] - y[i, j]) - \sum_{i=2}^N i \cdot (x[i, j] - x[i-1, j]))$
<b>Constraints:</b>
// $C_0$ to $C_3$ : Ensure sanity of the output $C_0 : x[i-1, j] \leq x[i, j], \forall (i, j), 2 < i < N, 1 < j < S$ $C_1 : y[i-1, j] \leq y[i, j], \forall (i, j), 2 < i < N, 1 < j < S$ $C_2 : x[i, j] \geq y[i, j], \forall (i, j), 1 < i < N, 1 < j < S$ $C_3 : y[i, j-1] \geq x[i, j], \forall (i, j), 1 < i < N, 1 < j < S$ // $C_4$ : Ensures that each individual section contains at least two markers of any type $C_4 : \sum_{i=1}^N (project[i] + client[i] + role[i] + teamsize[i] + description[i] + duration[i] + projectnum[i] + technology[i]) * (x[i, j] - y[i, j]) \geq 2 \cdot x[N, j], \forall j, 1 < j < S$ // $C_5$ : Ensures that each individual section contains at least one marker of $project, client, duration, projectnum$ $C_5 : \sum_{i=1}^N (project[i] + client[i] + duration[i] + projectnum[i]) * (x[i, j] - y[i, j]) \geq x[N, j], \forall j, 1 < j < S$ // $C_6$ to $C_{12}$ : Ensure that each individual section does not contain repeated markers of certain types $C_6 : \sum_{i=1}^N client[i] \cdot (x[i, j] - y[i, j]) \leq 1, \forall j, 1 < j < S$ $C_7 : \sum_{i=1}^N duration[i] \cdot (x[i, j] - y[i, j]) \leq 1, \forall j, 1 < j < S$ $C_8 : \sum_{i=1}^N teamsize[i] \cdot (x[i, j] - y[i, j]) \leq 1, \forall j, 1 < j < S$ $C_9 : \sum_{i=1}^N projectnum[i] \cdot (x[i, j] - y[i, j]) \leq 1, \forall j, 1 < j < S$ $C_{10} : \sum_{i=1}^N description[i] \cdot (x[i, j] - y[i, j]) \leq 1, \forall j, 1 < j < S$ $C_{11} : \sum_{i=1}^N project[i] \cdot (x[i, j] - y[i, j]) \leq 2, \forall j, 1 < j < S$ $C_{12} : \sum_{i=1}^N role[i] \cdot (x[i, j] - y[i, j]) \leq 2, \forall j, 1 < j < S$ // $C_{13}$ : Ensures that a project section follows a <i>blankline</i> or begins with <i>project, projectnum</i> or <i>duration</i> markers $C_{13} : (x[i, j] - x[i-1, j]) \leq (1 - (1 - blankline[i-1]) * (1 - projectnum[i]) * (1 - project[i]) * (1 - duration[i])), \forall (i, j), 2 < i < N, 1 < j < S$

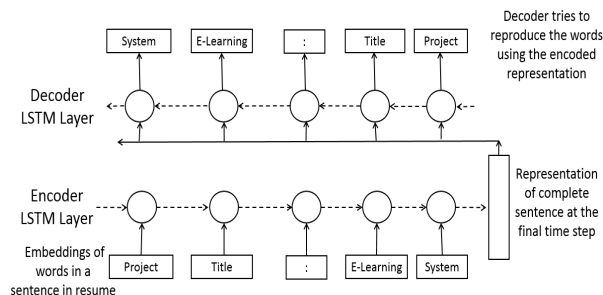
the Mean Squared Error at the decoder output. We train this model on a large corpus of 810 resumes having more than 140,000 sentences, using 100-dim word vectors and 200-dim vectors for sentence representations.

Using the encoder LSTM in our sequence autoencoder, we can now get an embedded representation for any new or unseen resume sentence. As discussed earlier, a resume is a sequence of sentences and our aim is to assign an appropriate label (from BIO labels) to each sentence to identify project sections.

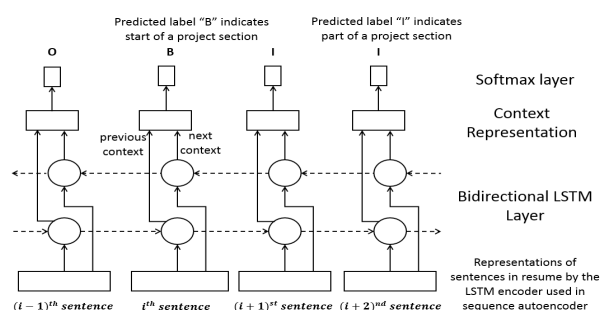
For this purpose, we design another Bi-directional LSTM model [7] where input for the Bi-LSTM layer is a sequence of embedded representations of sentences in a resume. For the  $i^{th}$  sentence, Bi-LSTM provides two representation vectors:

- previous* context vector which captures the context from the first to  $(i-1)^{th}$  sentences, and
- next* context vector which captures the context from  $(i+1)^{th}$  till the last sentence in the resume.

These two context vectors are concatenated and passed to a softmax layer for final prediction of BIO



**Fig. 1.** LSTM-based sequence autoencoder used to learn embedded representations for sentences in resumes



**Fig. 2.** Bi-LSTM sequence labelling model for assigning BIO labels to resume sentences

labels. The detailed architecture of this model is depicted in Figure 2.

This model is then trained in a supervised fashion (similar to CRF) using labelled dataset of resumes annotated with gold-standard project sections. Once the model is learned, it can be used for predicting BIO labels for any unseen resume and predicted project sections can be derived from these labels.

Moreover, we tried another variant of this Bi-LSTM based model, where for each sentence, we augment the sentence representation (provided by the LSTM encoder layer in our sequence autoencoder) with a vector representing the presence of project section markers.

This vector is a binary vector containing  $K$  bits corresponding to each project section marker. If any marker is present in a sentence, then its corresponding bits are set to 1, otherwise they are set to 0. The value of  $K = 10$  was empirically found to be suitable.

## 4 Experimental Evaluation

In this section, we describe the dataset details as well as intrinsic and extrinsic evaluation strategies.

### 4.1 Datasets

We manually annotated 366 resumes for project section information. For each resume, we identified sentence numbers corresponding to the first sentences of project sections. The dataset was partitioned into two parts:  $D_1$  (206 resumes) and  $D_2$  (160 resumes). The resumes in the dataset  $D_1$  were used to fine-tune the patterns for project markers as well as the rules used in knowledge-based approach. The supervised sequence labelling approaches (CRF and LSTM-based approaches) are trained on  $D_1$ . The dataset  $D_2$  is a blind test set for all approaches.

### 4.2 Intrinsic Evaluation

For intrinsic evaluation of the proposed approaches, we compare the predicted project sections with the manually annotated (gold-standard) project sections at 3 different evaluation levels (see Table 4):

- **Strict:** If first sentence number of a predicted section matches that of a gold-standard section, a true positive (TP) is counted. Unmatched predicted sections are counted as false positives (FP); unmatched gold-standard sections are false negatives (FN).
- **Lenient1:** If first sentence number of a predicted section is within  $\pm 3$  sentences of that of a gold-standard section, a TP is counted. A predicted section is counted as FP only if there is no gold-standard section starting within  $\pm 3$  sentences. A gold-standard section is counted as FN only if there is no predicted section starting within  $\pm 3$  sentences.
- **Lenient2:** When the first sentence number of a predicted section is within  $\pm 20$  words of that of a gold-standard section, a TP is counted. FPs/FNs are counted analogously.

Table 4 shows the comparative performance of the proposed approaches. All the approaches are evaluated on both the datasets  $D_1$  and  $D_2$ . For the

**Table 4.** Project section identification performance (Strict and Lenient Evaluations)

	Knowledge based			ILP			CRF			LSTM without markers			LSTM with markers		
	P	R	F	P	R	F	P	R	F	P	R	F	P	R	F
Strict- $D_1$	88.0	84.0	<b>86.0</b>	50.0	46.3	48.1	78.5	66.0	71.7	57.9	80.3	67.3	65.0	80.5	71.9
Lenient1- $D_1$	94.3	89.2	<b>91.7</b>	77.3	70.9	74.0	89.6	69.4	78.3	79.4	91.2	84.9	87.4	91.3	89.3
Lenient2- $D_1$	92.3	87.5	<b>89.8</b>	74.6	69.1	71.7	89.0	68.9	77.6	77.5	89.0	82.9	85.3	89.5	87.3
Strict- $D_2$	64.2	63.6	63.9	53.1	46.0	49.3	71.8	50.9	59.6	53.9	77.3	63.5	61.7	79.8	<b>69.6</b>
Lenient1- $D_2$	78.2	74.7	76.4	75.6	65.0	69.9	88.5	59.1	70.9	73.7	90.9	81.4	79.6	91.2	<b>85.0</b>
Lenient2- $D_2$	76.2	72.5	74.3	71.4	61.4	66.0	87.8	58.4	70.1	70.5	86.8	77.8	77.3	87.9	<b>82.2</b>

supervised approaches (CRF and LSTM-based), the dataset  $D_1$  is used for training the models which are then applied on the dataset  $D_2$ . Also, these approaches are also evaluated on  $D_1$  using 5-fold cross-validation. It can be observed that the knowledge-based approach outperforms all other approaches on  $D_1$ . But its performance degrades on  $D_2$  which is the blind test set.

Although such performance degradation is observed for all approaches, it is more pronounced for the knowledge-based approach. As the rules in the knowledge-based approach are designed by observing project sections in  $D_1$ , this approach seems to have over-fitted on  $D_1$ . Although, ILP-based approach lags behind the knowledge-based approach, its performance is more consistent across  $D_1$  and  $D_2$ . As ILP-based approach is a more principled way of representing the domain knowledge, it is also easier to maintain.

On the dataset  $D_2$ , the LSTM-based approach using markers outperforms all other approaches. Moreover, this approach provides more consistent performance across both the datasets  $D_1$  and  $D_2$ . Although, the CRF-based approach lags behind in F1, it provides the best precision on  $D_2$ . In future, we plan to explore an ensemble of CRF and LSTM-based approaches to exploit the high-precision nature of CRF-based approach as well as the high-recall nature of LSTM-based approach.

The LSTM-based approach without using markers also outperforms knowledge-based, ILP-based and CRF-based approaches. It is important to note that this approach has the least dependence on the domain knowledge because it does not need information about project section markers.

It only depends upon the sentence representations learned in an unsupervised manner using sequence autoencoder. Hence, the LSTM-based approach without markers can be easily re-trained for any other domain for identification of other types of repeated sections.

### 4.3 Extrinsic Evaluation

As mentioned earlier, several practical applications need project sections to be identified in resumes. So we used the resumes with or without identification of project sections to evaluate the output of some typical end-user queries. For example, a recruitment executive or project leader is interested in candidates who have used skill X in at least one project. Since the given skill X may occur outside project sections as well, one should only identify those candidates for whom X occurs within at least one project section. Not using project sections will retrieve all resumes in which X occurs *somewhere* in the resume, not necessarily within any project sections.

The results are shown in Table 5 for 3 different skills. In setting  $S_1$ , the resumes were retrieved without using any project section information. For settings  $S_2$  to  $S_6$ , the resumes were retrieved using the project section information predicted by knowledge-based approach, ILP-based approach, CRF-based approach, LSTM-based approach without markers and LSTM-based approach using markers, respectively. Using ground truth, we computed precision, recall and  $F$ -measure for these 6 settings. Other example queries where project sections are important are: *find candidates who have used skill X for at least 12 months* and *find candidates who have used skill X in role developer*.

**Table 5.** User queries with/without project sections

Query $\Rightarrow$ Setting $\Downarrow$	SQL Server 2008			Hibernate			Web Sphere		
	P	R	F	P	R	F	P	R	F
$S_1$ : No project sections	50.0	100.0	66.7	77.8	100.0	87.5	69.6	100.0	82.1
$S_2$ : Knowledge-based	100.0	100.0	100.0	100.0	100.0	100.0	93.3	87.5	90.3
$S_3$ : ILP	100.0	100.0	100.0	100.0	100.0	100.0	93.8	93.8	93.8
$S_4$ : CRF	50.0	100.0	66.7	100.0	71.4	83.3	100.0	75.0	85.7
$S_5$ : LSTM w/o markers	100.0	100.0	100.0	100.0	100.0	100.0	100.0	87.5	93.3
$S_6$ : LSTM with markers	100.0	100.0	100.0	100.0	100.0	100.0	100.0	93.8	96.8

## 5 Conclusions

In this paper, we proposed 2 unsupervised and 2 supervised methods for the novel problem of identifying repeated sections in a document (in particular, resumes). The proposed methods can also detect non-repeating sections containing specific types of information. Our knowledge-based method is interesting because it is modeled after the human ways of dealing with the same problem, but its drawback is that it is hard to maintain.

The ILP based method is similar but more robust. We compared the four methods on a dataset of resumes of IT professionals. The 2 supervised methods based on CRF and LSTM also perform well, where the LSTM-based method outperforms all other methods on our blind test set.

Though the CRF-based method underperforms the LSTM-based method, it achieves the highest precision among all the methods. In future, we plan to explore an ensemble of CRF and LSTM-based methods to exploit the high-precision nature of CRF-based method as well as the high-recall nature of LSTM-based method. We also evaluated the benefits of identifying such repeated sections in practical IR tasks. Topic-based section identification methods do not work well, because the same topics occur across different repeated sections.

The problem proposed here is of wider interest for fine-grained IR and can be used to identify sections in a wide variety of documents, such as legal documents, news, financial reports, scientific papers and web pages.

## References

1. Dai, A. M. & Le, Q. V. (2015). Semi-supervised sequence learning. *Advances in neural information processing systems*, pp. 3079–3087.
2. Garg, S., Singh, S. S., Mishra, A., & Dey, K. (2017). Cvbed: Structuring cvs using word embeddings. *IJCNLP: Volume 2*, volume 2, pp. 349–354.
3. Guo, Y., Reichart, R., & Korhonen, A. (2015). Unsupervised declarative knowledge induction for constraint-based learning of information structure in scientific documents. *TACL*, Vol. 3, No. 1, pp. 131–143.
4. Hirohata, K., Okazaki, N., Ananiadou, S., & Ishizuka, M. (2008). Identifying sections in scientific abstracts using conditional random fields. *IJCNLP: Volume-1*.
5. Hochreiter, S. & Schmidhuber, J. (1997). Long short-term memory. *Neural computation*, Vol. 9, No. 8, pp. 1735–1780.
6. Lafferty, J., McCallum, A., & Pereira, F. C. (2001). Conditional random fields: Probabilistic models for segmenting and labeling sequence data.
7. Lample, G., Ballesteros, M., Subramanian, S., Kawakami, K., & Dyer, C. (2016). Neural architectures for named entity recognition. *Proceedings of NAACL-HLT*, pp. 260–270.
8. Li, Y., Lipsky Gorman, S., & Elhadad, N. (2010). Section classification in clinical notes using supervised hidden markov model. *Proc. 1st ACM Int. Health Informatics Symposium*, pp. 744–750.
9. Pennington, J., Socher, R., & Manning, C. (2014). Glove: Global vectors for word representation. *Proceedings of the 2014 conference on empirical methods in natural language processing (EMNLP)*, pp. 1532–1543.

10. Shah, M., Palshikar, G., & Srivastava, R. (2017). New approaches of resume sectioning for automating talent acquisition. *Fifth Int. Conf. Business Analytics and Intelligence (ICBAI)*.
11. Singh, A., Rose, C., Visweswariah, K., Chenthamarakshan, V., & Kambhatla, N. (2010). Prospect: a system for screening candidates for recruitment. *CIKM*, pp. 659–668.
12. Yu, K., Guan, G., & Zhou, M. (2005). Resume information extraction with cascaded hybrid model. *ACL*, pp. 499–506.

*Article received on 18/01/2019; accepted on 04/03/2019.  
Corresponding author is Girish K. Palshikar.*

# Identifying Short-term Interests from Mobile App Adoption Pattern

Bharat Gaind\*, Nitish Varshney\*, Shubham Goel, Akash Mondal

Samsung Research Institute Bangalore,  
Bangalore, India

{bharat.gaind, nitish.var, shubham.goel, a.mondal}@samsung.com

**Abstract.** With the increase in an average user's dependence on their mobile devices, the reliance on collecting user's browsing history from mobile browsers has also increased. This browsing history is highly utilized in the advertising industry for providing targeted ads in the purview of inferring user's short-term interests and pushing relevant ads. However, the major limitation of such an extraction from mobile browsers is that browsing history gets reset when the browser is closed or when the device is shut down/restarted; thus rendering existing methods for identification of user's short-term interests on mobile devices, ineffective. In this paper, we propose an alternative method to identify such short-term interests by analysing user's mobile app adoption (installation/uninstallation) patterns over a period of time. Such a method can be highly effective in pinpointing the user's ephemeral inclinations like buying/renting an apartment, buying/selling a car or a sudden increased interest in shopping (possibly due to a recent salary bonus, he received). Subsequently, these derived interests are also used for targeted experiments. Our experiments result in up to 93.68% higher click-through rate in comparison to the ads shown without any user-interest knowledge. Also, up to 51% higher revenue in the long term is expected as a result of the application of our proposed algorithm.

**Keywords.** Word1, word2, word3.

## 1 Introduction

A decade-old advancement in the field of smartphones has made smartphones handy for every user. With the increase in an average user's dependence on their mobile devices, his dependence on traditional websites, typically browsed via personal computers, is decreasing. Consequently,

the focus of the advertising industry has also started to migrate from the web towards mobile devices. However, existing methodologies like cookies, which are used to provide a personalized experience or push targeted advertisements, are not applicable to the mobile ecosystem. The reason is that these cookies are reset, every time the browser is closed or when the phone is shut down/restarted. Additionally, they cannot be shared across apps. Therefore, such traditional methods make mobile marketers handicapped in knowing the user's interests.

To tackle this, the mobile advertisement industry, these days, is relying on Data Management Platform (DMP), which collects all of the user's cookies or events coming as first / second / third party data and infers the short-term interests of users. These interests are then used by Demand Side Platforms (DSP's) or advertising agencies to push targeted ads (formally known as creatives). However, not every advertiser agency or DSP has the luxury of using such DMPs, primarily due to cost restrictions.

In this paper, we propose an alternative method to identify the short-term interests of a user by carefully analysing their mobile app adoption (installation/uninstallation) patterns over a period of time and subsequently, supply these extracted interests to DSPs, that use them to suggest relevant ads for the user. This significantly increases the CTR (Click-Through Rate) of the suggested ads. Our method relies on the key realization that users constantly install and remove mobile-apps on need basis [6, 8], making this data stream valuable and a potential source to deduce their short-term needs, interests or inclinations.

\* Both authors have contributed equally.

Our system first collects the installed-apps list of a mobile-user in a repetitive interval and then extracts installation/uninstallation patterns from it. This installed-apps list contains the applications downloaded and installed by the user, some of which may be pre-installed by the device manufacturer. Features like app-description are then utilized to find the user's interests using various natural language processing (NLP) based unsupervised methods.

A shortcoming of such an approach is that periodic access to a user's installed apps might seem invasive to an individual's privacy. To avoid such an invasion, we have taken a number of measures. These measures include anonymizing user-identifiers end-to-end and collecting data of only those users who have given consent for data collection and analysis, which they often do to avail intelligent services offering enhanced features. We, therefore, answer all privacy-related questions raised by [10] and address this issue.

The rest of the paper is organized as follows. In Section 2, we discuss the related works and advancements that have been done in this area. We discuss both the relevant techniques as well as the commercial applications in this field. In Section 3, we give an overview of our system architecture, the modules and their functions. In Section 4, we give a detailed description of the datasets used, followed by results and discussions. In Section 5, we show how our proposed methodology would impact the revenue and CTR of an organization, if applied in the real world.

## 2 Related Work

Identifying a user's interests helps the advertising industry in pushing creatives that align with his preferences. Further, identifying his short-term interests allows the advertising industry to understand when and where the audience wants to engage, the content with which they want to interact, and then select the right place at the right time to push these creatives. This ultimately boosts the Click Through Rates (CTR) and Conversion Rate (CR) of the creatives displayed to the user [12].

In previous works, there are predominantly two concepts employed to identify a user's short-term interests. The first one explores recommendation based on search keywords, which are collected using cookies or events coming as first / second / third part-data and are used to infer the short-term interests of users [7]. The main limitation of cookies on mobile browsers is that they reset when the browser is closed or when the phone is shutdown / restarted [11]. Also, as discussed before, they cannot be shared across apps. Furthermore, such approaches cannot be applied to new users or users who like browsing in incognito mode, as such users data is not available with the marketers.

Another concept that is related to identifying the user's short-term interests is using a snapshot of his installed apps. To the best of our knowledge, research on identifying short-term interests from mobile app adoption pattern has not been conducted. However, some research on predicting user traits [15, 16] like gender, language, country, religion etc., has been done. These works, typically take a single snapshot of installed apps as input and use supervised learning to categorize users into their traits. Similarly, numerous studies have tried to find a user's traits, such as, whether the user is single, a parent, his mother tongue, the next app that he is going to install and his life-events [1, 8]. These studies predict a static user property (his traits). Recommending system models using only such user traits to identify user interests have lesser accuracy in comparison to the models using search keywords (discussed in the previous paragraph) [4]. Thus, existing methods that identifies user traits cannot be directly applied to identify the user's short-term interests.

Some of the commercial solutions related to identifying a user's interests are provided by Lotame<sup>1</sup>, Oracle BlueKai DMP<sup>2</sup>, Adobe Audience Manager<sup>3</sup> and Salesforce DMP (formerly Krux)<sup>4</sup>. These solutions typically organize the marketers'

---

<sup>1</sup><https://www.lotame.com/>

<sup>2</sup><https://www.oracle.com/marketingcloud/products/data-management-platform/index.html>

<sup>3</sup><https://www.adobe.com/in/analytics/audience-manager.html>

<sup>4</sup><https://www.salesforce.com/>



audience data into categories and taxonomies of user-interests and segment the audience to generate various insights.

### 3 Design and Algorithm

In this section, we explain how we calculate the short-term interests of a user, using his installed/uninstalled apps. Our aim is to retrieve semantically and linguistically important words related to a given app using the app's description (referred to as *app-description* from here on) available from google play store. Our approach consists of various methodologies, all of which are used independently to extract the most important words out of text. We use TF-IDF (Term Frequency-Inverse Document Frequency), YAKE (Yet Another Keyword Extractor), LDA (Latent Dirichlet Allocation), TextRank, TopicRank and Graph methods to achieve this, as shown in Fig. 1. We associate these important words, extracted out of the apps that the user installed over a period of time, as his short-term interests, over that period of time. Similarly, the important words extracted out of the apps that the user uninstalled are considered as events that the user is no longer interested in. All our methodologies are implemented in Python, using various inbuilt and external tools and libraries.

#### 3.1 Preprocessing

Preprocessing an app's description (mostly in the form of a paragraph, comprising of multiple sentences) consists of various steps. First, we remove stopwords using *nlk*<sup>5</sup> *corpus*. Next, we remove punctuations with the help of *string punctuations*. The third step is removing non-alpha-numeric characters (including non-English languages) using regex. We also find the POS (Parts Of Speech) tags of each of the words in every sentence in the original paragraph using *nlk*, and store these tags in a dictionary, for later use. Finally, we remove words, that occur very frequently across all app-descriptions using IDF (Inverse Document Frequency). The threshold

used for this step is 5%. In other words, all words that occur at least once in more than 5% of a randomly chosen app-descriptions set (of size 1000) are removed in this step. Also, before finding their frequency across all app-descriptions, we lemmatize all words in the app-description. This is done using *nlk*'s *WordNetLemmatizer*.

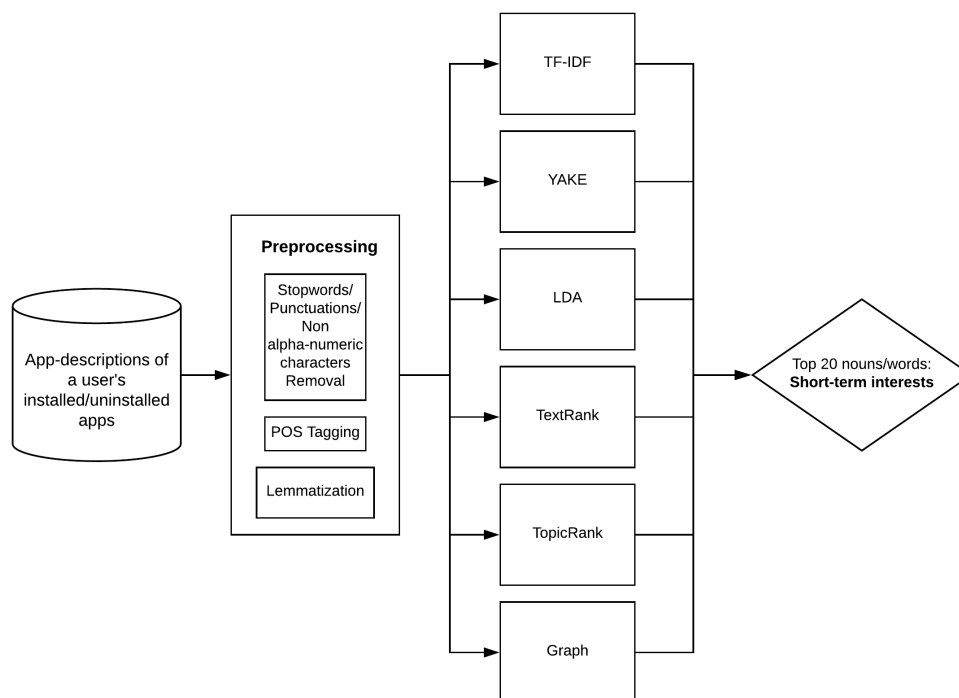
#### 3.2 Interest Identification using TF-IDF

The first methodology, we use is the standard TF-IDF (Term Frequency-Inverse Document Frequency) technique. It is used to find out the importance of each word in a collection of documents. After preprocessing, as explained above, we calculate TF by finding the frequency of each word and dividing it by the total number of words in the app-description. IDF is calculated by taking the logarithm of a fraction, whose numerator is the frequency of a word, across all app-descriptions, and the denominator is the total number of app-descriptions. Finally, the TF-IDF metric is calculated by multiplying the TF and IDF values. The output of this step is a metric of the importance of each word in the app-description. We then use the stored POS tags, calculated in the previous step, to filter out the nouns in the app-description and output the ones with the highest TF-IDF values.

#### 3.3 Interest Identification using YAKE

The next approach uses Yet Another Keyword Extractor (YAKE) [5], which is a statistical method for multi-lingual keyphrase extraction. Being an unsupervised method, YAKE avoids the problem of the long training process of other supervised methods and does not depend on any dictionaries for topic extraction. We implement the YAKE algorithm using the *pke* library [2] in python. After preprocessing the input text using the techniques mentioned in section 3.1, we load it as a document in a *pke* YAKE extractor instance. After this, we select 1-3 grams as keyphrase candidates and remove the candidates terminating with a stopword on either side. Next, the candidates are scored using various features extracted from the cleaned input such as word position (since most important words are usually in the beginning), word

<sup>5</sup>Natural Language Toolkit [http://www.nltk.org/], Version 3.3



**Fig. 1.** Steps involved in Short-term interest identification

frequency (the more the word occurs, the better) and other useful features defined in the YAKE algorithm. Finally, we choose 20 words with the lowest scores since they are the most important ones.

### 3.4 Interest Identification using LDA

In this approach, we use Latent Dirichlet Allocation (LDA), which is one of the most popular topic-modelling techniques and exploits the fact that every document is originally based on a combination of topics. It tries to backtrack and extract these constituent topics and also outputs a list of relevant words, corresponding to each topic. We use the *gensim* [14] library in python to implement LDA. First, we break down the app-description into a list of sentences, and then correspondingly, each sentence into a list of words. We use this corpus as input to *gensim* to create a dictionary of words, and subsequently,

a document-term matrix, which is then fed to the *LdaModel* to find topics. The number of topics selected is 1 (as usually each sentence in app description talks about 1 topic) and the number of passes is 50 (as app-description is usually small, therefore more passes were required). The output of this step is a list of 100 most important words in the app-description, sorted by their degree of associativity to the topic predicted. We then filter the top 20 nouns from these words (picked on the basis of their importance), using the POS tags stored in the preprocessing step and output them as the most important topics in the passage.

### 3.5 Interest Identification using TextRank

In this approach, we use TextRank [13], which is a graph-based method for keyword extraction. It implements the idea of voting of a vertex by its adjacent vertices. The vertex having the highest number of votes is the most significant vertex. The

score associated with a vertex is calculated using its own number of votes, and the score assigned to the vertices casting these votes. Again, we use the *pke* library in Python to implement this model. In this implementation, nodes are words of certain part-of-speech (nouns and adjectives) and edges represent the distance between word occurrences. Nodes are then ranked by the TextRank graph-based ranking algorithm. A window size of 3 is chosen and the top 5% vertices are used for phrase generation. Finally, the top 20 words with the highest weights are selected and output as the most important words in the text.

### 3.6 Interest Identification using TopicRank

In this approach, we employ TopicRank [3], which is an unsupervised and a graph-based keyphrase extraction method. Unlike TextRank, this method generates a graph of topics instead of words. Each topic is a collection of similar single and multi-word expressions. The advantage of generating a graph of topics is that the semantic relations between topics are better captured and the graph is more concise. To implement this method, we first preprocess the input data and load it into the *pke* library [2] TopicRank instance. Then, we select the important topics as groups of similar noun phrases and adjectives in the document using the inbuilt candidate selection method. Next, candidate ranking is done using random walk algorithm. Finally, the top 20 words with the highest weights are output.

### 3.7 Interest Identification using Graph

The final approach involves constructing a word co-occurrence network (or graph) using the app-description. Co-occurrence networks/graphs are the collective interconnections of terms based on their paired presence within a specified unit of text. Co-occurrence networks are generated by connecting pairs of terms using a set of criteria defining co-occurrence. Hence, the terms are nodes on the graph and the edges are co-occurrence between those terms. For us, it will be an occurrence in a similar sentence. The system goes through two phases for generating a co-occurrence network: the Graph construction

phase and selecting the top 20 nodes from the graph constructed.

In the graph construction phase, first, we split the app-description into sentences. For each sentence, we define co-occurrence between words, which corresponds to an edge between the co-occurrence words. We defined three types of co-occurrence between a pair of words. According to the first type, two words co-occur if they occur in the same sentence. The second type implies co-occurrence when two words occur in the same sentence and are neighbors. Lastly, the third co-occurrence means two words occurring in the same sentence and being at a max distance of two. For example, in the sentence “stock market is booming high”, < “stock”, “high” > and < “stock”, “booming” > co-occur by definition 1. Also, < “stock”, “market” > co-occur by definition 2 and 3. In order to realize all such possibilities, we generated three separate graphs based on different co-occurrences. We also removed all self-loops present in the graph.

Next, we consider words (nodes) from our co-occurrence graph using various graph metrics. Various graph metrics considered for experimentation are *Degree*, *Page Rank*, *Betweenness centrality* and *Closeness centrality*. We took the top 20 words, based on the aforementioned graph metrics. All the graph operations including construction and metric calculations are done using *networkX* [9] in python.

## 4 Results

As mentioned before, we take the top 20 most important words as the output of each of the methodologies explained above. In this section, we explain how we evaluate these methodologies. Our project work in the ads domain allows us to access the current apps installed in the user's Android phone. Using this data, we find all installed and uninstalled apps of users during a fixed period of time (July 1 to July 15, 2018, in our case). We then apply each approach explained above to find the 20 most important words out of the app-descriptions of each of these installed/uninstalled apps. These words act as short-term interests (or dislikes, in the case of

uninstalls) of the user. Our evaluation methods compute an estimator of the potential increase in CTR that could be generated by the application of this new methodology. The experiments are conducted on a dataset of real users in an offline manner after imitating a real-time environment. During imitation, we have considered all eligible bids (including losing bids in the ad-bidding process) for each advertisement request. Each bid object contains the advertiser's unique identifier, the advertised product description and bidding price.

Now, we need a dataset that acts as ground truth to assess our output. Another dataset available to us, as part of our project, is the CTR-dataset (Click Through Rate-dataset) of around 1 million users. We use it to identify the clicks and the impressions of the ads shown to the user from July 7 to July 21, 2018. The reason this period of time is chosen is that we assume that on an average, the short-term interests of the user in the period July 1 to July 15, will be at their peak on July 7th.

Therefore, the probability of him clicking an ad of the similar category (as his installed apps) will be the highest during July 7th to July 21st (chosen because his short-term interests might persist for around 7 more days after he's finished installing/uninstalling apps between July 1st and July 15). Since we consider the event that a user clicks on an ad of a particular category as an indicator of his short-term interests, this dataset can act as ground-truth.

Next, we calculate the CTR ( $CTR_1$ ) for all creatives (or ads) that are similar to his installed/uninstalled apps. This similarity is calculated by finding the top 20 words for both the ads and the installed apps, using the various approaches discussed above, and then using *nltk wordnet* to find synonyms (Two apps are similar if their app-descriptions have at least two common words/synonyms).  $CTR_1$  represents what the CTR would have been if the creatives were chosen according to the results predicted by our methodologies. We also calculate CTR ( $CTR_2$ ) of the ads that are dissimilar to the user's installed apps.  $CTR_2$  represents the probability of a user clicking an ad if our approach is not used.

The evaluation is based on the hypothesis that for a particular user and his short-term interests identified based on installed apps,  $CTR_1 > CTR_2$  (since if our methodologies are used, the CTR is expected to increase, as the ads now have become more personalized/interesting to the user). Similarly, for interests identified based on uninstalled apps, our hypothesis is  $CTR_1 < CTR_2$  (since our methodologies are used, in this case, to identify ads that are not of interest to the user anymore). The average  $CTR_1$  and  $CTR_2$  over 34,908 users for install-based interest identification and 28,226 users for uninstall-based interest identification, computed this way is depicted in Table 1 and 2. The CTR increase column denotes how much the CTR increased from  $CTR_2$  to  $CTR_1$  in case of Table 1, and  $CTR_1$  to  $CTR_2$ , in case of Table 2.

Additionally, our algorithm is not applicable to all the bids. (For some bids, the top keywords identified did not match with any of the user's installed/uninstalled apps' top keywords). Hence, we have also tabulated the percentage of bids (denoted as applicable bids), for which different methods were applicable. As can be observed from Table 1, the TextRank model shows the maximum CTR increase of 93.68% for interests identified based on installed apps, but its applicability is low. Similarly, In Table 2, the Closeness Centrality ranking method of type 2 shows the maximum CTR increase of 113.58% for uninstallation-apps based non-interest identification, with a relatively low applicability. To increase the applicability while maintaining the increase in CTR, we have implemented various priority-based Hybrid models, comprising of 3 models having the maximum increase in CTR and one model having high applicability. In a hybrid model, we try to find the common keywords between a user's identified interests (installs/uninstalls) and the incoming bids, using the the highest-priority model (LDA, for instance, in case of installs).

If this model fails, the next model is tried out to find these common words and so on. Table 3 and 4 denote the various hybrid models implemented. Each model comprises of 4 constituent models as discussed above. The

**Table 1.** CTR enhancement for interests identified based on installed apps

Method		CTR1 (%age)	CTR2 (%age)	Increase in CTR (%age)	Applicable Bids (%age)
TF-IDF		4.99	3.37	48.07	61
YAKE		4.09	3.57	14.57	64
LDA		5.28	2.79	<b>89.20</b>	40
TextRank		5.52	2.85	<b>93.68</b>	24.6
TopicRank		5.34	3.28	<b>62.8</b>	64.7
Degree graph ranking	Co-occ. type 1	4.52	3.69	22.49	78.8
	Co-occ. type 2	4.51	3.49	29.23	78.6
	Co-occ. type 3	4.64	3.42	<b>35.67</b>	<b>80.6</b>
PageRank graph ranking	Co-occ. type 1	4.38	3.87	13.18	79.9
	Co-occ. type 2	4.51	3.50	<b>28.85</b>	<b>81.3</b>
	Co-occ. type 3	4.50	3.50	28.57	79.8
Betweenness centrality ranking	Co-occ. type 1	4.61	3.29	39.74	77.6
	Co-occ. type 2	4.71	3.26	<b>44.48</b>	<b>78.8</b>
	Co-occ. type 3	4.66	3.29	41.64	78.4
Closeness centrality ranking	Co-occ. type 1	4.52	3.79	19.26	76.6
	Co-occ. type 2	4.52	3.86	17.09	80.3
	Co-occ. type 3	4.71	3.31	<b>42.29</b>	<b>82.1</b>

**Table 2.** CTR enhancement for interests identified based on uninstalled apps

Method		CTR1 (%age)	CTR2 (%age)	Increase in CTR (%age)	Applicable Bids (%age)
TF-IDF		3.13	3.98	27.16	<b>75.89</b>
YAKE		3.22	4.53	40.68	72.21
LDA		3.54	4.76	34.46	<b>85.26</b>
TextRank		3.29	5.15	56.53	74.00
TopicRank		3.23	4.71	45.82	<b>78.84</b>
Degree graph ranking	Co-occ. type 1	2.92	5.80	98.63	60.42
	Co-occ. type 2	2.95	5.83	97.62	60.63
	Co-occ. type 3	2.91	5.89	<b>102.41</b>	<b>63.89</b>
PageRank graph ranking	Co-occ. type 1	2.95	5.62	90.51	60.03
	Co-occ. type 2	2.89	5.93	<b>105.19</b>	<b>63.26</b>
	Co-occ. type 3	2.91	5.88	102.61	66.03
Betweenness centrality ranking	Co-occ. type 1	2.93	5.77	96.92	60.03
	Co-occ. type 2	2.94	5.83	98.30	62.63
	Co-occ. type 3	2.96	5.51	86.15	59.47
Closeness centrality ranking	Co-occ. type 1	2.98	5.97	100.33	60.63
	Co-occ. type 2	2.87	6.13	<b>113.58</b>	<b>68.06</b>
	Co-occ. type 3	2.94	6.11	107.82	60.21

hybrid model, *LDA\_TopicRank\_TextRank\_Degree-3*, for instance, consists of the LDA, TopicRank, TextRank and the degree graph (Co-occurrence

type 3) models. A significant improvement can be seen in the applicability of the bids (85.6% and 82%, respectively) in the hybrid models, while

**Table 3.** CTR enhancement for interests identified based on installed apps using priority-based Hybrid models

Hybrid Method	CTR1 (%age)	CTR2 (%age)	Increase in CTR (%age)	Applicable Bids (%age)
LDA_TopicRank_TextRank_ Degree-3	6.822	3.674	85.65	85.6
LDA_TopicRank_TextRank_ PageRank-2	6.773	3.701	82.97	86.8
<b>LDA_TopicRank_TextRank_ BwCent-2</b>	6.934	3.647	<b>90.13</b>	<b>85.6</b>
LDA_TopicRank_TextRank_ CICent-3	6.772	3.671	84.47	85

**Table 4.** CTR enhancement for interests identified based on uninstalled apps using priority-based Hybrid models

Hybrid Method	CTR1 (%age)	CTR2 (%age)	Increase in CTR (%age)	Applicable Bids (%age)
Degree-3_PageRank-2_ CICent-2_TF-IDF	3.239	6.698	106.75	79.6
Degree-3_PageRank-2_ CICent-2_LDA	3.365	7.085	110.52	88.6
<b>Degree-3_PageRank-2_ CICent-2_TopicRank</b>	3.277	7.27	<b>121.82</b>	<b>82</b>

maintaining an impressive CTR increase (90.13% and 121.82%, respectively), as shown in Table3 and 4.

## 5 Impact on Revenue

In the advertising industry, there are majorly three players from a pricing perspective. The first ones are the advertisers who want to acquire new users via advertisement. Second, there are publishers who have a dedicated space where the advertisement can be shown to the user, formally known as an ad inventory. And lastly, there are intermediaries (mostly DSPs) making the match via the bidding process. The advertisers usually want to pay for per-user action (a click or an app install), which is formally known as Cost per Action (CPA) and the publishers want to earn per impression, formally known as Cost per Mile (CPM) impression.

Hence, the intermediaries also need to perform arbitrage. We define arbitrage as a process of converting CPA to CPM. The prevalent

methodologies applied for performing arbitrage predict the CTR for each bid, which can then be used as  $CPA \times CTR$  to get CPM (assuming intermediary is not taking any cut out of it). For example, consider the case when an advertiser has an ad budget of \$10 per click and the intermediary predicted that the CTR for the advertisement is 0.1 for a user's request. In such a scenario, the intermediary may bid for \$1 ( $\$10 \times 0.1$ ) for an ad space and the advertisement would be selected for display, if \$1 is the highest-priced valid bid received.

From the last decade, publishers also like to provide targeted advertisements, even at the cost of initial revenue loss, as it improves the long-term value of their brand by not cluttering their ad space. In the process, an initial revenue loss would likely result as the highest-priced bid might not be the most liked bid by the user (identified using our methodology).

This initial revenue loss is tabulated in Table 5, where we have shown the initial revenue impact on publishers after applying various methodologies proposed in this paper.

**Table 5.** Real time impact on application of our algorithm

<b>Method</b>		<b>Initial Impact (%age)</b>	<b>Increase in CTR (%age)</b>	<b>Applicable Bids (%age)</b>	<b>Estimated Long-term Impact(%age)</b>
TF-IDF		-20.1435	48.07	61	18.60584
YAKE		-13.6853	14.57	64	21.83496
LDA		-22.5013	89.2	40	<b>37.21169</b>
TextRank		-23.1164	93.68	24.6	<b>51.102</b>
TopicRank		-17.9908	62.8	64.7	<b>29.1184</b>
Degree graph ranking	Co-occ. type 1	-13.1727	22.49	78.8	7.842132
	Co-occ. type 2	-9.48232	29.23	78.6	13.12148
	Co-occ. type 3	-8.14967	35.67	80.6	<b>17.68324</b>
PageRank graph ranking	Co-occ. type 1	-13.4803	13.18	79.9	5.894413
	Co-occ. type 2	-9.53357	28.85	81.3	<b>11.48129</b>
	Co-occ. type 3	-9.78985	28.57	79.8	11.22501
Betweenness centrality ranking	Co-occ. type 1	-10.0974	39.74	77.6	17.32445
	Co-occ. type 2	-15.223	44.48	78.8	15.42799
	Co-occ. type 3	-9.53357	41.64	78.4	<b>19.52845</b>
Closeness centrality ranking	Co-occ. type 1	-13.1727	19.26	76.6	9.431061
	Co-occ. type 2	-13.3778	17.09	80.3	5.996929
	Co-occ. type 3	-11.0712	42.29	82.1	<b>17.99077</b>

**Table 6.** Long-term impact on publishers revenue

Part A : Without application of our algorithm

	<b>Advertiser's ad budget per click</b>	<b>Intermediary's predicted CTR</b>	<b>Publisher's revenue per impression</b>
Advertiser1	\$12	0.1	$\$12 \times 0.1 = \$1.2$
Advertiser2	\$10	0.1	$\$10 \times 0.1 = \$1.0$

Part B : On application of our algorithm on the second advertiser's bid

	<b>Advertiser's ad budget per click</b>	<b>Intermediary's predicted CTR</b>	<b>Publisher's revenue per impression</b>
Advertiser1	\$12	0.10000	$\$12 \times 0.10000 = \$1.2000$
Advertiser2	\$10	0.14467	$\$10 \times 0.14467 = \$1.4467$

In this table, we have computed the revenue loss on 1 million user requests containing an ad inventory of three different publishers. For computing the estimated long-term impact, the increase in CTR (%age) and applicable users (%age) metrics have been taken from Table 1.

In Table 6, through an example, we try to explain how the revenue has increased in the long-term for the publishers in Table 5. We consider two

advertisers, having an ad budget of \$12 and \$10 per click respectively, who want to get 100 clicks each for their advertisements.

The intermediary has set the default CTR (to convert CPA to CPM) for both the advertisements as 0.1 (in the absence of any system to identify user interests).

Additionally, assume that the publisher has an ad inventory of 1000. Therefore, the publisher

will earn \$1200 ( $=\$1.2 \text{ cost per impression} * 1000 \text{ impressions}$ ), when our system is not in place.

Now, suppose that the second ad is more aligned to the user's interests (as predicted by our algorithm), as a result of which it is chosen by the publisher instead of the first one. In this case, the publisher will initially earn \$1000, which would mean an initial loss of \$200 for every 1000 impressions. However, in the long term, the second advertiser's target of reaching 100 clicks would be achieved after only 691 impressions (assuming a 44.67% average increase in CTR and 100% applicability from Table 5). In this case, the predicted CTR would be 0.14467, which is why the number of impressions needed would be 100 clicks divided by 0.14467, which comes to approximately 691 impressions. Since the goal of the second advertiser (of achieving 100 clicks) has been achieved in fewer than 1000 impressions, it will not participate anymore in the bidding process, which leaves the ad space for 309 impressions available to the publisher. The publisher can then show the first advertiser's ad in this ad space, which will generate an additional revenue of \$371 ( $=309 \text{ impressions} * \$12 \text{ per click} * 0.1 \text{ predicted CTR}$ ).

Hence, the total revenue of  $\$1000 + \$371$  would be earned by the publisher, which is \$171 more than the revenue earned when our system was not deployed. Since, in real-world scenarios, publishers display millions of impressions per month, applying our algorithm could have a significant impact on the long-term revenue of the publishers.

## 6 Conclusion and Future Work

In this paper, we propose various methodologies to identify the short-term interests of a user by analysing his mobile app adoption (installation/uninstallation) patterns over a period of time. Such a method can be highly effective in pinpointing the user's ephemeral inclinations. Our experiments result in around 94% higher click-through rate (in case of installed-apps based interest identification using TextRank algorithm) and around 113% higher click-through rate (for uninstalled-apps based non-interest identification using the closeness-centrality ranking method of

graphs), in comparison to the ads shown without any user-interest knowledge.

Further, we implement several hybrid models having both a high CTR increase and bid-applicability (as high as 121.82% and 82%, in the dislike-identification case). Also, around 51% higher revenue in the long-term is expected as a result of the application of our proposed algorithm. In future, we would optimize our methodologies to decrease their execution-times, since our priority in this paper was achieving a higher CTR increase.

Also, we would work on unifying various installation and uninstallation-based models to make the overall system better personalized to the user's interests.

## References

1. Baeza-Yates, R., Jiang, D., Silvestri, F., & Harrison, B. (2015). Predicting the next app that you are going to use. *Proceedings of the eighth ACM international conference on web search and data mining*, ACM, pp. 285–294.
2. Boudin, F. (2016). pke: an open source python-based keyphrase extraction toolkit.
3. Bougouin, A., Boudin, F., & Daille, B. (2013). Topicrank: Graph-based topic ranking for keyphrase extraction.
4. Burke, R. (2007). Hybrid web recommender systems. In *The adaptive web*. Springer, pp. 377–408.
5. Campos, R., Mangaravite, V., Pasquali, A., Jorge, A. M., Nunes, C., & Jatowt, A. (2018). A text feature based automatic keyword extraction method for single documents. *European Conference on Information Retrieval*, Springer, pp. 684–691.
6. Chittaranjan, G., Blom, J., & Gatica-Perez, D. (2013). Mining large-scale smartphone data for personality studies. *Personal and Ubiquitous Computing*, Vol. 17, No. 3, pp. 433–450.
7. Elmeleegy, H., Li, Y., Qi, Y., Wilmot, P., Wu, M., Kolay, S., Dasdan, A., & Chen, S. (2013). Overview of turn data management platform for digital advertising. *Proceedings of the VLDB Endowment*, Vol. 6, No. 11, pp. 1138–1149.
8. Frey, R. M., Xu, R., & Ilic, A. (2015). Reality-mining with smartphones: Detecting and predicting life events based on app installation behavior: Research-in-progress. *ICIS 2015 Proceedings*.



9. Hagberg, A., Schult, D., & Swart, P. (2005). Networkx: Python software for the analysis of networks. *Mathematical Modeling and Analysis, Los Alamos National Laboratory*.
10. Jedrzejczyk, L., Price, B. A., Bandara, A. K., Nuseibeh, B., Hall, W., & Keynes, M. (2009). I know what you did last summer: risks of location data leakage in mobile and social computing. *Department of Computing Faculty of Mathematics, Computing and Technology The Open University*, pp. 1744–1986.
11. Lele, S. (2014). Cookies in mobile: Do they exist. <https://www.socialmediatoday.com/content/cookies-mobile-do-they-exist>.
12. Merisavo, M., Vesanen, J., Arponen, A., Kajalo, S., & Raulas, M. (2006). The effectiveness of targeted mobile advertising in selling mobile services: an empirical study. *International Journal of Mobile Communications*, Vol. 4, No. 2, pp. 119–127.
13. Mihalcea, R. & Tarau, P. (2004). Texttrank: Bringing order into text. *Proceedings of the 2004 conference on empirical methods in natural language processing*, pp. 404–411.
14. Rehurek, R. & Sojka, P. (2010). Software framework for topic modelling with large corpora. *Proceedings of the LREC 2010 Workshop on New Challenges for NLP Frameworks*, Citeseer.
15. Seneviratne, S., Seneviratne, A., Mohapatra, P., & Mahanti, A. (2014). Predicting user traits from a snapshot of apps installed on a smartphone. *ACM SIGMOBILE Mobile Computing and Communications Review*, Vol. 18, No. 2, pp. 1–8.
16. Seneviratne, S., Seneviratne, A., Mohapatra, P., & Mahanti, A. (2015). Your installed apps reveal your gender and more! *ACM SIGMOBILE Mobile Computing and Communications Review*, Vol. 18, No. 3, pp. 55–61.

Article received on 20/01/2019; accepted on 04/03/2019.  
Corresponding author is Nitish Varshney.



# Joint Learning of Named Entity Recognition and Dependency Parsing using Separate Datasets

Arda Akdemir<sup>1</sup>, Tunga Güngör<sup>2</sup>

<sup>1</sup> University of Tokyo, Tokyo,  
Japan

<sup>2</sup> Bogazici University, Istanbul,  
Turkey

aakdemir@hgc.jp, gungort@boun.edu.tr

**Abstract.** Joint learning of different NLP-related tasks is an emerging research field in Machine Learning. Yet, most of the recent models proposed on joint learning require a dataset that is annotated jointly for all the tasks involved. Such datasets are available only for frequently used languages. In this paper, we propose a novel BiLSTM CRF based joint learning model for dependency parsing and named entity recognition tasks, which has not been employed before for Turkish to the best of our knowledge. This enables joint learning of various tasks for languages that have limited amount of annotated datasets. Our model, tested on a frequently used NER dataset for Turkish, has comparable results with the state-of-the-art systems. We also show that our proposed model outperforms the joint learning model which uses a single dataset.

**Keywords.** Joint learning, named entity recognition, dependency parsing, Turkish.

## 1 Introduction

Named Entity Recognition (NER) is the task of detecting and categorizing the entities in a given text. Entities contain valuable information related to various Natural Language Processing (NLP) tasks which makes NER an important and a popular research area.

Nadeau et al. [14] give a detailed survey of the work done until 2007 in this area. Many of the systems relied on manually crafted feature sets and used statistical machine learning methods to make NER predictions.

Most feature based models require third party tools, like morphological analyzers, to annotate a given dataset for the features [16]. Recent state-of-the-art works focus on representing each word with character and word embeddings [11], and learn the entity tags by using Bidirectional Long Short Term Memory (BiLSTM) layers. One of the main motivations of these approaches is to relieve NLP models from relying on manually crafted features.

Even though these systems do not rely on manually crafted feature sets, for joint learning of multiple tasks these systems require a dataset to be annotated jointly for all the tasks involved. However, such jointly annotated datasets usually do not exist, especially for less frequently used languages such as Turkish.

In this paper, we propose a joint learning model that attempts to solve this joint annotation problem. Our model jointly learns named entity recognition and dependency parsing using separate datasets for each task similar to the approach used for named entity recognition and morphological disambiguation [6]. Dependency parsing information is shown to improve the performance of feature based NER systems. Based on this observation, we incorporate dependency parser output into the named entity recognition component to learn both tasks jointly.

Our results show that joint learning on separate gold-labeled datasets for each task outperforms

joint learning on a single dataset annotated automatically using a third party tool [15] for the dependency parsing tags. Moreover, our model is the first model in the literature that attempts joint learning of dependency parsing and named entity recognition for Turkish language. Our main contributions can be summarized as follows:

- We implement a novel joint learning model for dependency parsing and named entity recognition.
- We propose a novel way of learning these tasks where we make use of different datasets for each task.

The paper is organized as follows: Section 2 describes the related work on NER and joint learning. Section 3 describes the datasets used for each task. Section 4 explains the details of the proposed joint learning model. Section 5 gives the results obtained. Section 6 concludes the paper. All the work we have done can be accessed and reproduced from the relevant GitHub repository which includes the source codes, the results obtained and explanation about how the model should be run<sup>1</sup>.

## 2 Related Work

Most of the recent work on NER make use of neural network models and especially BiLSTM based systems. Chiu et al. [3] use a BiLSTM based neural network model and learn the character embeddings using Convolutional Neural Networks (CNNs). Lample et al. [11] propose a system based on BiLSTMs and Conditional Random Fields (CRFs) where CRFs are used to find the optimal tag sequence using the Viterbi decoding algorithm. Ma et al. [13] propose a model that combines the previous two works. They use CNNs to learn character embeddings and use BiLSTMs together with a CRF layer. Their model currently holds the state-of-the-art result for the NER task on the frequently used CoNLL 2003 English dataset with 91.21% F1 score.

<sup>1</sup><https://github.com/ardakdemir/Named-Entity-Recognition-in-Turkish-Using-Deep-Learning-Models-and-Joint-Learning>

Related work for Turkish, which make use of manually crafted feature sets, show that NER performance increases when syntactical and morphological features are employed [16]. Their findings show the importance of using features for morphologically rich languages like Turkish. Following the state-of-the-art work for English, Gungor et al. [6] use a BiLSTM and CRF based model, and show that morphological features improve the NER performance. Using a similar BiLSTM based deep learning architecture, Gunes et al. [5] obtained the state-of-the-art score for the commonly used NER dataset for Turkish [18].

Joint learning is an emerging research area in NLP. Joint learning of POS tagging and dependency parsing is shown to improve the dependency parser performance [15, 12] for the CoNLL 2018 Shared Task: Multilingual Parsing from Raw Text to Universal Dependencies. Syntactic parsers are used to increase the performance of various NLP tasks. However, these models are in pipeline format and are not trained jointly. Hashimoto et al. [7] attempt joint learning of various NLP tasks of different levels of complexity. They follow the work of Sogard et al. [17] which shows using outputs of different levels of the neural network for different tasks outperforms using the outputs at the same level for each task. Similarly, Gungor et al. [6] showed that joint learning of morphological disambiguation and named entity recognition improves the named entity recognition performance when different layer outputs are used for each task. Recently, Katiyar et al. [8] proposed a joint-learner for the extraction of entity mentions and relations for English language. Bekoulis et al. [1] also proposed an attention-based joint learner for entity and relation extraction.

## 3 Datasets

This section describes the datasets used in this paper. For the named entity recognition task for Turkish we used a frequently used NER dataset [18]. The dataset was created as part of a work which tackles four information extraction tasks including NER. It is made up of articles from a national newspaper. Table 1 gives the number of entities in each set.

**Table 1.** Number of annotated entities in the Turkish NER dataset

Subset	LOC	ORG	PER
Training	6,720	9,260	6,249
Development	769	1,412	824
Test	907	1,174	670

For the configuration of our model which uses a single dataset annotated for both tasks, we used a dependency parser to annotate this dataset for dependency parsing.

The IOB tagging scheme is used for the NER task. The dataset is annotated for three entity types: Location, Organization and Person. Including the label of the non-entity words we have the following seven entity tags: B-PER, I-PER, B-LOC, I-LOC, B-ORG, I-ORG, and O. In the IOB scheme, each entity is labeled with the 'I' prefix unless the entity token is an immediate successor of a separate entity token with same entity type. In that case, 'B' prefix is used to overcome the entity boundary ambiguity problem.

For dependency parsing we used the IMST-UD dataset provided by the Universal Dependency framework for the CoNLL 2018 Shared Task [19]. The dataset is in CoNLL-U format which was designed to form a universal format for dependency datasets of multiple languages. The dataset is a semi-automatic conversion of the IMST Treebank, which is itself a reannotated version of the METU-Sabanci Turkish Treebank. The dataset is made up of 5,635 sentences from daily news reports and novels. An example annotated sentence from this dataset is given in Fig. 1.

The example sentence in Turkish is: 'Karşısında, pantolonu dizlerine dek ıslak, önlük torbası ham eriklerle dolu İbrahim dikiliyordu', which corresponds to the following English sentence: 'İbrahim was standing against him with his pants wet up to the knees and with his bag filled with plums.' The joint learner only makes use of the surface form, dependency relation type, and dependency head index fields which correspond to 2<sup>nd</sup>, 7<sup>th</sup>, and 8<sup>th</sup> fields, respectively.

## 4 Methodology

This section explains in detail the joint learning model we propose. The joint learning model is a BiLSTM CRF based neural network model that has two main components corresponding to the two tasks being tackled. We begin by explaining the overall model which will be followed by detailed descriptions of each component in separate sections.

The overall architecture of the model is shown in Fig. 2. The first component in the network produces the output for the dependency parsing task. Given a token sequence, BiLSTM is used to calculate the vector representation of each word. These vectors are given to two separate Multilayer Perceptrons (MLP) which output the scores for the directed dependency arcs between each pair of tokens and the relation type prediction for the predicted dependency arcs, respectively. The second part of the network is responsible for the named entity recognition task. It takes as input the output produced by the dependency parsing component in vector format, in addition to the same inputs given to the dependency parser. BiLSTM is used to find the vector representation of each token, by taking into account the prediction of the dependency parser. These vectors are given to a MLP which outputs scores for all possible NER tags for each token. A CRF layer is used to find the highest scoring tag sequence for the calculated tag scores.

### 4.1 Dependency Parsing Component

The dependency parsing component of our proposed model is similar to the dependency parsing component of the joint POS Tagging Dependency Parsing (jPTDP) system [15]. Word, character and capitalization embeddings of the tokens of a given sentence are given as input to the dependency parser. Following the related works [3], in addition to the word and character representations, we feed the model with the vector representation of the capitalization feature of each token. Capitalization information is important for the NER task. BiLSTMs are used to produce vector representations of each token by analyzing the

**Fig. 1.** An example sentence from the IMST-UD Treebank dataset used by the dependency parser of the joint learner

the LSTM layer has length  $ldims$  for each direction, thus the output is of length  $2 \times ldims$ . The second BiLSTM layer thus takes as input a vector of size  $2 \times ldims$  for each token in a sentence. The system again goes over these vector representations to create forward and backward vector outputs of size  $ldims$ .

The outputs of the second BiLSTM layer are used as the input for the multi layer perceptrons (MLP) responsible for calculating the scores for the dependency parsing task. The dependency parsing task contains two sub-tasks: creating a parse tree for a given sentence and labeling the arcs of the parse tree.

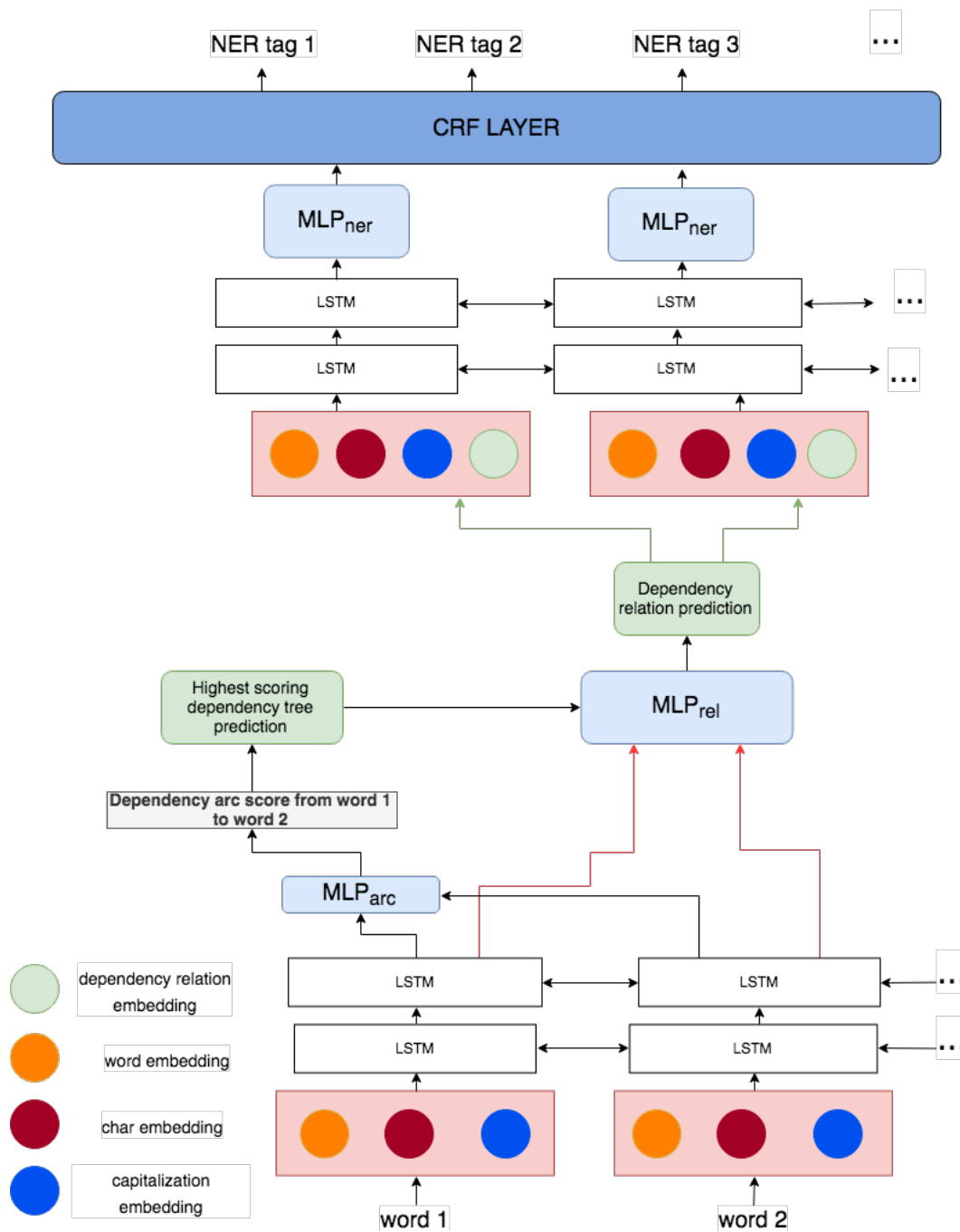
$\mathbf{c}_{emb}$  is calculated by using a BiLSTM layer. For a character  $x$ , we randomly initialize a character embedding  $\mathbf{c}_x$ .

Let  $e_{w_i}^{lstm}$  represent the final lstm output for word  $w_i$ . Following the related work [15], four vectors are concatenated and given as input to the MLP called **MLP**<sub>arc</sub>, which outputs the score for a directed arc from  $w_i$  and  $w_j$ :

$$\mathbf{c}_{emb} = \mathbf{c}_{emb}^f \circ \mathbf{c}_{emb}^b.$$

$$\mathbf{e}_w = \mathbf{w}_{emb} \circ \mathbf{c}_{emb}^f \circ \mathbf{c}_{emb}^b \circ \mathbf{cap}_{emb}.$$

where  $(\mathbf{e}_{w_i}^{lstm} * \mathbf{e}_{w_j}^{lstm})$  and  $|\mathbf{e}_{w_i}^{lstm} - \mathbf{e}_{w_j}^{lstm}|$ , are element-wise multiplication and absolute element-wise difference, respectively. Each vector is of length  $2 \times ldims$  which results in a vector of size  $8 \times ldims$ . Given these scores Eisner’s decoding algorithm [4] is used to find the highest scoring dependency tree. Loss of this sub-task,  $Loss_{arc}$ , is calculated by maximizing the difference between the gold parse tree and the highest scoring incorrect parse tree following the related work [9].



**Fig. 2.** Architecture of the joint learning model for dependency parsing and named entity recognition

To find the dependency relation type, i.e. the label, for each predicted arc, another MLP called  $\mathbf{MLP}_{rel}$  is used.  $\mathbf{MLP}_{rel}$  takes the same input with the previous MLP and outputs a vector containing a score for each relation type:

$$\text{scores}_{rel}(i, j) = \mathbf{MLP}_{rel}(\mathbf{e}_{w_i}^{lstm} \circ \mathbf{e}_{w_j}^{lstm} \circ (\mathbf{e}_{w_i}^{lstm} * \mathbf{e}_{w_j}^{lstm}) \circ |\mathbf{e}_{w_i}^{lstm} - \mathbf{e}_{w_j}^{lstm}|). \quad (2)$$

Cross entropy loss,  $Loss_{rel}$ , is computed over this score vector using the gold label for each token.

## 4.2 Named Entity Recognition Component

Let  $rel_j$  represent the dependency relation type for the dependency arc from  $w_i$  to  $w_j$ . Each dependency relation type  $rel_j$  is represented with an embedding  $\mathbf{e}_{rel_j}$ . Given the relation type prediction  $\mathbf{e}_{rel_j}$  of the dependency parsing component for given words  $w_i$  and  $w_j$ , the word  $w_j$  is represented by concatenating the embedding of this relation to the vector representation of the word:

$$\mathbf{e}_{w_j}^{ner} = \mathbf{e}_{w_j} \circ \mathbf{e}_{rel_j}.$$

For a given input sentence with  $n$  words, we represent each sentence with the sequence of vector representations  $\mathbf{e}_{w_i}^{ner}$  for  $1 \leq i \leq n$  and feed this sequence of vectors into LSTMs in forward and backward directions. The LSTM outputs vectors  $\mathbf{v}_i$  for each word  $w_i$  in a given sentence by taking into account the context in both directions:

$$\mathbf{v}_i = \text{LSTM}_f(\mathbf{e}_{w_{1:i}}^{ner}) \circ \text{LSTM}_b(\mathbf{e}_{w_{n:i}}^{ner}).$$

These vectors of size  $2 \times ldims$  are fed into a second LSTM layer which outputs the final vector representation of each token:

$$\mathbf{v}_k^{fin} = \text{LSTM}_f(v_{1:k}) \circ \text{LSTM}_b(v_{n:k}).$$

Each vector  $\mathbf{v}_k^{fin}$  is given as input to an MLP called  $\mathbf{MLP}_{ner}$  which produces scores for each possible entity type for each word. The score matrix of size  $(n, t)$  is created where  $score_{i,j}$  refers to the score for the  $i^{th}$  token having the  $j^{th}$  tag:

$$\text{Scores}_{ner}(i) = \mathbf{MLP}_{ner}(\mathbf{v}_i^{fin}).$$

During the prediction mode, these scores are normalized into probabilities to be used for finding the optimal tag sequence. For each tag for a given word probability is calculated by normalizing the scores produced for each entity tag type  $tag_j$  using the softmax function:

$$P(i, j) = \frac{\exp(\text{score}(i, j))}{\sum_{(j' \in \text{tags})} \exp(\text{score}(i, j'))},$$

where  $\text{score}(i, j)$  represents the score produced for  $tag_j$  for a given word  $w_i$  in a sentence.

We model the tag sequence jointly rather than predicting each label independently. For this a CRF [10] is used to find the highest scoring named entity tag sequence. Transitions between named entity tags are important because of the sequential nature of the task and CRFs are used frequently for the NER task [6, 2]. Using the score vectors produced for each word and randomly initialized tag transition probabilities, we find the optimal tag sequence using the Viterbi decoding algorithm.

Negative log likelihood loss  $Loss_{ner}$  is used to calculate the loss of the gold label NER tag for each word in the sentence. The transition probabilities between entity tag types are included implicitly in  $Loss_{ner}$  because the final prediction of the model is calculated using the Viterbi algorithm which takes into account the transition probabilities.

## 5 Experiments and Results

We performed various experiments with different versions of the proposed joint learning model. This section will explain the experiments conducted and the results obtained. In this paper we mainly focus on experiments with the following configurations:

- Model 1: The named entity recognition component only. This configuration does not take as input the dependency parsing prediction. It is a typical BiLSTM CRF based named entity recognition model.
- Model 2: The joint learning model using a single dataset annotated jointly for both tasks. The named entity recognition dataset is automatically annotated using a third party dependency parsing tool [15].



- Model 3: Our proposed novel joint learning model which uses different datasets for each task. Extensive results are obtained for this final proposed model with various hyperparameter configurations. The results obtained with this model are compared with the results obtained in the previous two configurations.

For the evaluation of the performance of both of the tasks, we used the frequently used evaluation metrics. For the dependency task, 'Labeled Attachment Score' (LAS) and 'Unlabeled Attachment Score' (UAS) are used. For the named entity recognition task, we use precision, recall and F1-Measure.

We first show the results on the development sets used to fine tune the models. These results are calculated taking into account partial matches of entities so every match between a gold label and a prediction is counted without checking whether a named entity is completely predicted by the system or not. Fig. 3 shows the results obtained for running Model 1 for 40 epochs. The best F1-Measure of **0.915** is obtained in the 34<sup>th</sup> epoch. As stated, the architecture of this model is the same as shown in Fig. 2, but the part of the neural network responsible for the dependency parsing task is not used. One can think of this model as the dependency part of the network 'silenced'.

The next experiment is conducted for Model 2 on the version of the dataset that is annotated jointly for both tasks using a third party dependency parsing tool. The best results obtained for each task and the epoch numbers are given in Table 2 where DEP refers to the dependency parser score and NER refers to the named entity score. The dependency parser results for Model 2 are significantly higher than Model 3 (See Table 3). This is probably due to the fact that the architecture of the tool used for annotating the dataset is quite similar to the dependency parser implemented in our system. This similarity can make it easier for the system to quickly learn and mimic the parser used. The results also show that we could not obtain a performance improvement over Model 1.

Next, we performed extensive experiments for Model 3. First, we give the results for the default

configuration of the model on the development sets of each dataset. The parameters of the default configuration are set using the previous works on joint learning [6, 15]. The results with the default configuration are given in Table 3. The results show a relative improvement for the NER performance over using a single dataset annotated automatically.

There are various parameters in our joint learning model. The parameters of the model are given in Table 4. The results obtained for various combinations of word embedding dimensions and LSTM dimensions for Model 3 are given in Table ??.

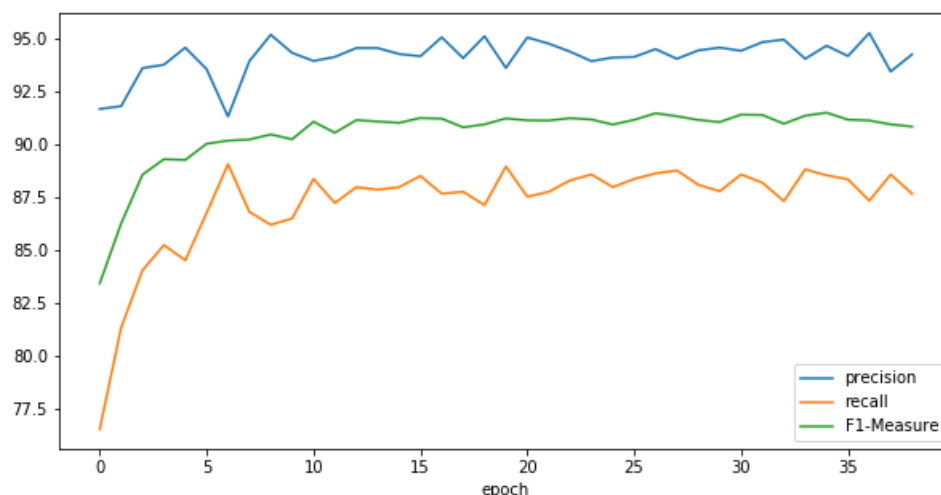
We could not observe a significant difference in performance during the grid search of other parameter configurations so we do not include them here. The best F1-Measure scores are obtained for the NER task with the following two configurations: (word embedding dim: 100 , lstm dim: 64) and (word embedding dim: 150 , lstm dim: 64).

Finally, we give the results obtained on the NER test set. We use the CoNLL evaluation metric for consistency with other works on named entity recognition. Table 6 shows the results for all models.

The results given here show that the novel joint learning model (Model 3) brings a relative improvement over using a single dataset annotated jointly for both tasks (Model 2) for all entity types. Yet, we could not observe an improvement over the NER only model. The overall F1-Measure for Model 1, Model 2, and Model 3 are 89.21, 82.11 and 84.56, respectively. On the other hand, the joint learning model has several advantages over the NER only model. It enables learning both tasks jointly and, with sufficient amount of training time, Model 3 has the potential to match the Model 1 performance on the NER task while learning an additional task.

## 6 Conclusion and Future Work

In this paper, we proposed a novel neural network model for joint learning, which attempts to solve the joint annotation problem. We combined two BiLSTM based neural components to jointly learn



**Fig. 3.** Results for the NER only model in each epoch tested on the development set

**Table 2.** Best results for the joint learning model on a single dataset

Task Name	Metric	Best Results	Best Epoch
DEP	Average LAS - UAS Accuracy	0.760	13
NER	F1-Measure	0.878	12

**Table 3.** Best results for the joint learning model on development sets for each task

Task Name	Metric	Best Results	Best Epoch
DEP	Average LAS - UAS Accuracy	0.600	16
NER	F1-Measure	0.904	17

**Table 4.** Parameters of the joint learning model together with the default values

Parameter Name	Default Value
Word embedding size	100
Character embedding size	50
Capitalization feature embedding size	50
Relation embedding size	100
Hidden units	100
Activation function	tanh
Lstm layers	2
Lstm dimensions	128
Enable dependency parsing	True
Enable viterbi decoding	True

dependency parsing and named entity recognition on different datasets. Our results show that we can obtain improvements over using a single dataset for joint learning. Yet, the NER component trained without using the dependency parsing prediction outperforms both models.

As future work, we plan to improve the performance of the proposed system in several ways. Using embedding representations for outputs instead of the outputs directly is shown to increase the performance of NLP systems for various tasks. Representation learning is applied to the coarse-grained labeled NER task in various studies. Using output representations of NER labels increases the performance for

**Table 6.** Results for all three models

	Model 1			Model 2			Model 3		
	Precision	Recall	F1	Precision	Recall	F1	Precision	Recall	F1
<b>PER</b>	89.74	89.89	89.81	92.43	77.82	84.50	86.29	86.66	86.48
<b>LOC</b>	89.95	90.04	89.99	77.07	87.53	81.97	86.84	85.89	86.36
<b>ORG</b>	87.56	86.65	87.10	81.21	75.67	78.34	80.97	76.41	78.63
<b>Overall</b>	89.28	89.15	89.21	83.78	80.51	82.11	85.23	83.91	84.56

both coarse-grained and fine-grained NER tasks. Future work includes learning the representations of output labels to calculate losses in a more robust way.

We will also modify the proposed architecture to take into account the dependency prediction of the dependency parsing component in different ways. We will implement a joint learning model that makes use of the head of the dependency arc for a given word as well as the dependency relation.

The following ideas should be mentioned:

- Use syntactic relations for embeddings?
- Use attention to pick which words to take into account during the learning of embeddings.
- Use subword level embeddings (morphological embeddings).
- Use BERT outputs.

## Acknowledgements

We would like to thank Onur Gungor for his kind help and advices about this work. This work was partially supported by JST CREST Grant Number JPMJCR1402, JSPS KAKENHI Grant Numbers 17H01693, and 17K20023JST.

## References

1. Bekoulis, G., Deleu, J., Demeester, T., & Devellder, C. (2018). Joint entity recognition and relation extraction as a multi-head selection problem. *Expert Systems with Applications*, Vol. 114, pp. 34–45.
2. Chen, T., Xu, R., He, Y., & Wang, X. (2017). Improving sentiment analysis via sentence type classification using bilstm-crf and cnn. *Expert Systems with Applications*, Vol. 72, pp. 221–230.
3. Chiu, J. P. & Nichols, E. (2015). Named entity recognition with bidirectional lstm-cnns. *arXiv preprint arXiv:1511.08308*.
4. Eisner, J. M. (1996). Three new probabilistic models for dependency parsing: An exploration. *Proceedings of the 16th conference on Computational linguistics-Volume 1*, Association for Computational Linguistics, pp. 340–345.
5. Güneş, A. & Tantuğ, A. C. (2018). Turkish named entity recognition with deep learning. *2018 26th Signal Processing and Communications Applications Conference (SIU)*, IEEE, pp. 1–4.
6. Güngör, O., Üsküdarlı, S., & Güngör, T. (2018). Improving named entity recognition by jointly learning to disambiguate morphological tags. *arXiv preprint arXiv:1807.06683*.
7. Hashimoto, K., Xiong, C., Tsuruoka, Y., & Socher, R. (2017). A joint many-task model: Growing a neural network for multiple nlp tasks. *EMNLP*.
8. Katiyar, A. & Cardie, C. (2017). Going out on a limb: Joint extraction of entity mentions and relations without dependency trees. *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 917–928.
9. Kiperwasser, E. & Goldberg, Y. (2016). Simple and accurate dependency parsing using bidirectional lstm feature representations. *arXiv preprint arXiv:1603.04351*.
10. Lafferty, J., McCallum, A., & Pereira, F. C. (2001). Conditional random fields: Probabilistic models for segmenting and labeling sequence data.
11. Lample, G., Ballesteros, M., Subramanian, S., Kawakami, K., & Dyer, C. (2016). Neural architectures for named entity recognition. *arXiv preprint arXiv:1603.01360*.
12. Li, Z., He, S., Zhang, Z., & Zhao, H. (2018). Joint learning of pos and dependencies for multilingual universal dependency parsing. *Proceedings of the*

- CoNLL 2018 Shared Task: Multilingual Parsing from Raw Text to Universal Dependencies*, pp. 65–73.
13. **Ma, X. & Hovy, E. (2016).** End-to-end sequence labeling via bi-directional lstm-cnns-crf. *arXiv preprint arXiv:1603.01354*.
  14. **Nadeau, D. & Sekine, S. (2007).** A survey of named entity recognition and classification. *Lingvisticae Investigationes*, Vol. 30, No. 1, pp. 3–26.
  15. **Nguyen, D. Q. & Verspoor, K. (2018).** An improved neural network model for joint pos tagging and dependency parsing. *arXiv preprint arXiv:1807.03955*.
  16. **Şeker, G. A. & Eryiğit, G. (2012).** Initial explorations on using crfs for turkish named entity recognition. *Proceedings of COLING 2012*, pp. 2459–2474.
  17. **Søgaard, A. & Goldberg, Y. (2016).** Deep multi-task learning with low level tasks supervised at lower layers. *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, volume 2, pp. 231–235.
  18. **Tür, G., Hakkani-Tür, D., & Oflazer, K. (2003).** A statistical information extraction system for turkish. *Natural Language Engineering*, Vol. 9, No. 2, pp. 181–210.
  19. **Zeman, D., Hajič, J., Popel, M., Potthast, M., Straka, M., Ginter, F., Nivre, J., & Petrov, S. (2018).** CoNLL 2018 shared task: Multilingual parsing from raw text to universal dependencies. *Proceedings of the CoNLL 2018 Shared Task: Multilingual Parsing from Raw Text to Universal Dependencies*, Association for Computational Linguistics, Brussels, Belgium, pp. 1–21.

Article received on 14/01/2019; accepted on 04/03/2019.  
Corresponding author is Arda Akdemir.

# Kazakh Text Summarization using Fuzzy Logic

Altanbek Zulkhazhav<sup>1</sup>, Zhanibek Kozhirbayev<sup>1,2</sup>, Zhandos Yessenbayev<sup>2</sup>, Altynbek Sharipbay<sup>1</sup>

<sup>1</sup> L.N.Gumilyov Eurasian National University,  
Faculty of Information Technologies,  
Kazakhstan

<sup>2</sup> Nazarbayev University,  
National Laboratory Astana,  
Kazakhstan

{altinbekpin, sharalt}@gmail.com, {zhanibek.kozhirbayev,zhyessenbayev}@nu.edu.kz,

**Abstract.** In this paper we present an extractive summarization method for the Kazakh language based on fuzzy logic. We aimed to extract and concatenate important sentences from the primary text to obtain its shorter form. With the rapid growth of information on the Internet there is a demand on its efficient and cost-effective summarization. Therefore the creation of automatic summarization methods is considered as a very important task of natural language processing. Our approach is based on the preprocessing of the sentences by applying morphological analysis and pronoun resolution techniques in order to avoid their early rejections. Afterwards, we determine the features of the processed sentences need for exploiting fuzzy logic methods. Additionally, since there is no available data for the given task, we collected and manually annotated our own dataset from the different Internet resources in the Kazakh language for the experimentation. We also applied our method on CNN/Daily Mail dataset. The ROUGE-N indicators were calculated to assess the quality of the proposed method. The ROUGE-L(f-score) score by the proposed method with pronoun resolution for the former dataset is 0.40, whereas for the latter one it is 0.38.

**Keywords.** Extractive text summarization, natural language processing, fuzzy logic.

## 1 Introduction

With the rapid growth of information on the Internet, it becomes extremely difficult for users to get what they really intend. Therefore, the creation of automatic summarization methods is considered as a very important task of natural

language processing. This allows the user to quickly understand large amount of information. Automatic text summarization is a task to extract the most important part of the source text in a shorter way. It can be performed in two ways: extraction and abstraction. The extraction method selects sentences or phrases that have high marks of importance, and combines them into a new short text, without changing the selected units. In the method of abstraction, the main content is extracted from the source text and paraphrased using linguistic methods for semantic analysis and text interpretation.

In this work we experiment with extractive summarization method for the Kazakh language based on fuzzy logic. We collected and manually annotated our own dataset from the different Internet resources in the Kazakh language. Additionally, we conduct experiment on CNN/Daily Mail dataset [7], which is an open dataset for use in text summarization experiments in English. Our approach is based on the preprocessing of the sentences by applying morphological analysis and pronoun resolution techniques. Afterwards, we determine features to extract important sentences from the text. The architecture of the extractive text summarization approach based on fuzzy logic is shown in Figure 1.

This paper is organized as follows: Section II presents a brief review of the related works. Section III presents a methodology of proposed techniques for automatic text summarization. More precisely, it describes the following stages: text

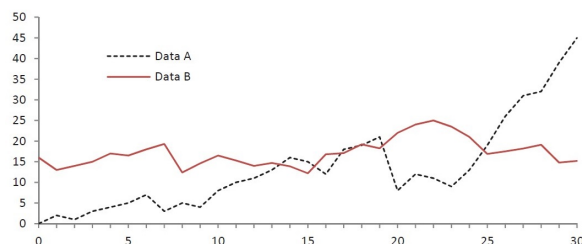


Fig. 1. Fuzzy logic system architecture for extractive text summarization

preprocessing, feature extraction and calculation, summarization using fuzzy logic. The dataset collection and experimental setup are described in Section IV. Experiment details and obtained results of the extractive text summarization task are also presented in this section. Summary of the performed experiments and areas of further research are given in Section V.

## 2 Related Work

This Section presents a brief review about automatic text summarization techniques. Many different techniques have been proposed for this task that utilizes a variety of different approaches. For a thorough review of works on text summarization the reader is advised to consult a very recent survey by [1].

We limit ourselves to a brief review of extractive text summarization, as our main aim is to score and select text units which have highest scores as summary. Shallow features [3], hidden markov models [8], discourse structure model [9], maximum marginal relevance [6], fuzzy logic [23] and swarm intelligence [4], conceptual graphs [5,18] approaches are proposed to deal with this task.

A lot of research has been done with respect to the Kazakh language [10–12, 14, 16, 20], but there are a few researches regarding automatic text summarization. [19] implemented and compared different summarization techniques based on TextRank algorithm, namely: General TextRank, BM25, LongestCommonSubstring. They conducted experiments on corpora of news articles parsed from the web written in Russian and Kazakh. [22]

performed an experiment to summarize articles from online news websites [tengrinews.kz](http://tengrinews.kz) with extractive way.

## 3 Methodology

### 3.1 Text preprocessing

In this work we experiment with news articles of the most popular Kazakhstani news websites. To prepare our collected data set for the summarization task, we preprocess the text by deleting unnecessary indents, spaces, punctuation marks and other specific characters as described in [13,21]. Afterwards, we perform a segmentation similar to [2], which involves a breakdown of the text into sentences and tokenization of each sentence. The next steps in the preprocessing pipeline include such tasks as lemmatization, numerals identification, named entity recognition and finding pronouns. For these we compiled a morphologically dictionary and devised the empirical rules. Finally, we developed a rule-based algorithm for the pronoun resolution which for each found pronoun basically scans several previous sentences and calculates the most probable word or phrase it refers to. This is necessary to improve the quality of summarization, since pronouns such as “I”, “he”, “she”, “they” usually are referred to “stop words” and, thus, removed from the text in the early stages. However, they indicate specific persons and carry certain significance. As a result of morphological and syntactic analysis, pronouns were replaced with the names of the persons they indicate. The pseudo-code of the algorithm used to pronoun replacement is illustrated in Figure 2. We remove stop words that are often found in the text, but do not represent a special meaning for determining the importance of the content. Deletion of affixes from a word by stemming concludes the text preprocessing.

### 3.2 Feature Extraction and Calculation

After preprocessing the text, it is necessary to extract features and calculate the functions of the sentence, the results of which are vectors of seven elements for each sentence. The elements of each

---

```

Input: text
Result: text with proper nouns in place of pronouns
for sentence in text.sentences:
  for word in sentence.words
    if isPronoun(word):
      antecedents = findAllAntecedentsFromText(text, word)
      candidate = chooseMostSuitableCandidate(antecedents)
      word = replace (word, candidate)
    end
  end
end
return text

```

---

Fig. 2. Algorithm 1: Pronoun replacement

vector take values in the interval  $[0, 1]$ . We consider the following features:

- Title feature (F1): It is defined as a ratio of the number of matches of the Title words (Tw) in the current sentence (S) to the number of words (w) of the Title (T) [24]:

$$F1(S) = \frac{\text{Number of Tw in } S}{\text{Number of words in } T}. \quad (1)$$

- Sentence Length (F2): It is defined as a ratio of the number of words (w) in the current sentence (S) to the number of words in the longest sentence (LS) in the text [24]:

$$F2(S) = \frac{\text{Number of } w \text{ in } S}{\text{Number of } w \text{ in } LS}. \quad (2)$$

This function is necessary for filtering from the selection of short and incomplete sentences, such as an author of the article, date of the article, etc.

- Sentence position (F3): It is defined as a maximum of the next two relations [24]:

$$F3(S) = \max \left( \frac{1}{\text{Position of } S}; \frac{1}{\text{Number of } S - \text{Position of } S + 1} \right). \quad (3)$$

If the sentence is at the beginning of the text, then the first expression is the maximum, if the sentence is at the end of the text, then the maximum value will be taken by the second expression. This function is important when selecting, as more informative sentences are usually located at the beginning or at the end of the text.

- Thematic word (F4): It is defined as a ratio of the number of thematic words (Thw) in the current sentence (S) to the maximum number of thematic words (Thw) calculated on all sentences (S) of the text [24]:

$$F4(S) = \frac{\text{Number of Thw in } S}{\max(\text{Number of Thw in all } S)}. \quad (4)$$

Thematic words are the most frequently used words in the text. They are directly related to the main theme of the text. We chose the five most frequent words in the text as thematic ones.

- Term Weight (F5): It is defined as a ratio of the sum of the frequencies of term occurrences (TO) in a sentence (S) to the sum of the frequency of term occurrences in the text [24]:

$$F5(S) = \frac{\sum (\text{Frequency of TO in } S)}{\sum (\text{Frequency of TO in all } S)}. \quad (5)$$

To calculate the weight of a sentence, we find the frequency with which the term appears in the sentence and the frequency with which the same (current) term appears in the text

- Proper Noun (F6): It is defined as a ratio of the number of proper nouns (PN) in a sentence (S) to the length (L) of a sentence [24]:

$$F6(S) = \frac{\text{Number of PN in } S}{L \text{ of } S}. \quad (6)$$

Proper nouns found in the proposal carry a lot of information about personal facts. Therefore, the sentences with the most proper nouns are an important part of the content.

- Numerical Data (F7): It is defined as a ratio of the number of numerical data (ND) in the sentence (S) to the length (L) of the sentence [24]:

$$F7(S) = \frac{\text{Number of ND in } S}{L \text{ of } S}. \quad (7)$$

Typically, numerical data has specific important values for summarization. Therefore, numerical data in the text could not be skipped.

The importance of sentences regarding features is presented in Table 1.

Table 1. Importance of sentences (rule examples)

Features	Low	Medium	High
topic/title	poor	average	decent or good
thematic word	poor	average	decent or good
term freq.	poor	average or mediocre	decent or good
proper noun	-	-	good
numerical data	-	-	not(poor)
sentence length	-	-	not(poor)
sentence position	-	-	not(poor)

### 3.3 Fuzzy Logic System Design

Fuzzy logic system design includes the following concepts: fuzzy set, membership function, fuzzy logic operations, linguistic variables, linguistic terms, fuzzy logical values, fuzzy logic conclusion [26]. A typical fuzzy logic system consists of the following components:

- fuzzifier;
- logical conclusion on the base of fuzzy knowledge;
- defuzzifier.

The fuzzifier determines a correspondence between the clear numerical value of the input variable and the value of the membership function of the corresponding term of the linguistic variable. In our case, the linguistic variables are the seven functions defined by us above. They take meanings from a variety of words such as “poor”, “mediocre”, “average”, “decent”, “good”. These words are called term-sets and take values in the interval  $[0,1]$  (Figure 3). In short, fuzzification is the process of transition from a clear to a fuzzy representation [26].

The fuzzifier depends on the membership function for the corresponding linguistic terms. One of the main problems of using fuzzy logic is a choice of the membership functions of the linguistic variables. The main types of the membership functions are triangular, trapezoidal, piecewise linear, Gaussian, sigmoid, and other functions. The choice of the membership function of a particular variable is a poorly formalized problem, the solution of which is based on intuition and experience [26]. For our task, we have prepared a more appropriate

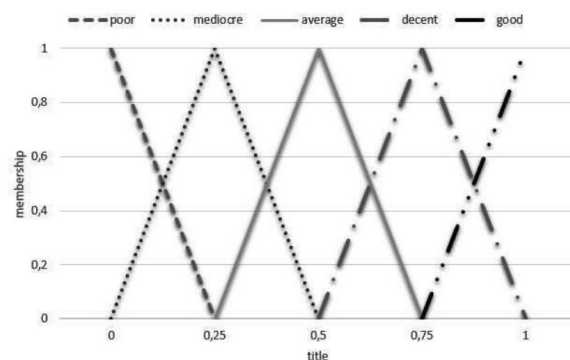


Fig. 3. Linguistic variable “Title feature”

triangular membership function used to specify uncertainties of the type: “approximately equal”, “average value”, “located in the interval”, “similar to the object”, “similar to the object”, etc.

The quality of fuzzy inference depends on the correct construction of “IF-THEN” rules. We obtained rules for a fuzzy knowledge base on the basis of analysis of manually written summaries. Since all the membership functions of linguistic variables are known to us, and the rules we need are defined, we proceed to the aggregation process. Aggregation is a procedure for determining the truth degree of conditions according to each rules of the fuzzy inference system. The values of the membership functions of the linguistic variable terms obtained at the stage of fuzzification are used. If the condition of a fuzzy production rule is a simple fuzzy statement, then the degree of its truth corresponds to the value of the membership function of the corresponding term of the linguistic variable. If a condition is a compound statement, then the truth degree of the complex statement is



determined on the basis of the known truth values of its elementary statements using the fuzzy logic operations introduced earlier [26]. After the logical inference, we obtain fuzzy values by accessing the fuzzy knowledge base. Also we obtain clear values for the output using the defuzzification of the fuzzy values of the linguistic variables (Figure 1).

Defuzzification is the procedure for converting a fuzzy set to a clear number. In the theory of fuzzy sets, the defuzzification is similar to finding the position characteristics (expectation, mode, median) of random variables in probability theory. The simplest way to perform the defuzzification is to select a clear number corresponding to the maximum membership function [26].

For software implementation of text summarization based on fuzzy logic, we used the python language and the skfuzzy package [25]. We constructed the membership function for each function value from five fuzzy sets: poor, mediocre, average, decent, good. Example of the membership function of the header function (Figure 3)

The last step of the fuzzy inference is defuzzification, i.e. output membership function, which we have broken into three: low, medium, high (Figure 4). The pseudo-code of the algorithm used to pronoun replacement is illustrated in Figure 5.

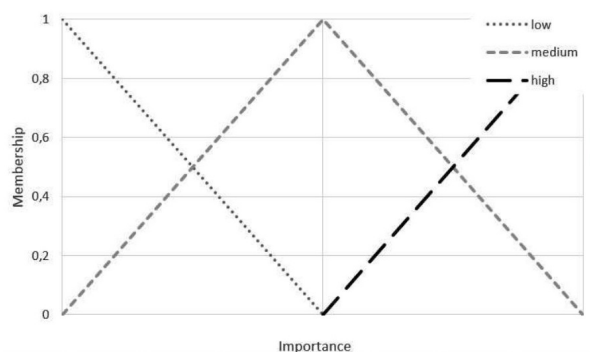


Fig. 4. Linguistic variable “Title feature”.

## 4 Experimental Setup and Results

### 4.1 Data Set

The data set for the given task was collected from the news articles of the most popular Kazakhstani

online news websites, namely kt.kz, bnews.kz, qazaquni.kz and qazaqtimes.com. The articles cover a wide range of topics and hence represent styles with high variety. Human annotators were asked to write an extractive summary of the article with respect to the style it was written in. Moreover, since we utilized the ROUGE package evaluation metric [15] which uses the reference summary or ideal summary, the extractive summary pair has to be verified by at least two annotators. The professional activity of each annotator also has to be taken into account as well. We also assess the performance of our approach on the selected part of CNN/Daily Mail dataset, which is a popular and free dataset for use in text summarization experiments. This dataset consists of news articles paired with multi-sentence summaries. For the pronoun resolution we used Stanford CoreNLP toolkit [17].

The average number of sentences in articles and the average number of sentences in summaries for both dataset are presented in Table 2.

### 4.2 Results

In this section, we present our experimental results for the automatic text summarization. We compare results obtained through applying our approach on both the Kazakh news dataset and CNN/Daily Mail dataset.

ROUGE metrics were used for a preliminary assessment of the work quality. More precisely, ROUGE-L considers sentence level structure similarity and determines the longest co-occurring in sequence n-grams in automatic way. ROUGE-1 shows the overlap of unigram between the system and reference summaries, whereas ROUGE-2 indicates for bigrams [15].

Table 3 lists the results of the experiments on text summarization for the Kazakh news dataset. As it can be seen the proposed pronoun resolution achieves better result rather than without pronoun resolution. The significant increase is indicated for Rouge-L (f-score): from 0.35 to 0.40. Rouge-1 (f-score) and Rouge-2 (f-score) scores are risen to 0.02 and 0.03, respectively. Moreover, we applied our method to CNN/Daily Mail dataset. Rouge-1 (f-score) and Rouge-L (f-score) scores show the

---

```

Input: source text
Result: summary of text
sentences = getSentence(text)
for sentence in sentences:
    tokens = doLinguisticAnalysis(sentence)
    features.add(getTopicFeature(tokens))
    features.add(getSentencePosition(tokens))
    features.add(getSentenceLength(tokens))
    features.add(getTFFeature(tokens))
    features.add(getThematicFeature(tokens))
    features.add(getProperNounFeature(tokens))
    features.add(getNumeralsFeature(tokens))
for feature in features:
    fuzzyCalculationInputs.add(doFuzzification(feature))
end
importanceValue = doFuzzyCalculationsByRules(fuzzyCalculationInputs)
importantPropertyOfSentences.add(importanceValue)
end
numberOfSummarySentences = max(sentences.count*0.3, 3)
return doDefuzzifier(first numberOfSummarySentences with maximum value)

```

---

Fig. 5. Algorithm 2: Summary extraction algorithm

Table 2. Data set characteristics

Data set	Number of articles	Average number of sentences in articles	Average number of sentences in summaries
Kazakh news	100	14	4.1
CNN/Daily Mail	100	39	3.75

Table 3. Rouge scores for the Kazakh news dataset

Rouge	metrics	Without pronoun resolution	With pronoun resolution
Rouge-1	precision	0.38	0.39
	recall	0.39	0.44
	f-score	0.36	0.38
Rouge-2	precision	0.32	0.34
	recall	0.33	0.37
	f-score	0.31	0.34
Rouge-L	precision	0.38	0.39
	recall	0.40	0.44
	f-score	0.35	0.40

same result, which are 0.38, whereas Rouge-2 (f-score) shows slightly worse result. An example of the automatic text summarization for the Kazakh language is shown in Table 5.

## 5 Conclusion

The research in the field of computational linguistics for the Kazakh language is expanding rapidly. Therefore, the results of the work will be very popular for a quick public perception of the

Table 4. Rouge scores for the CNN / Daily Mail dataset with pronoun resolution

Rouge	metrics	CNN/Daily Mail dataset
Rouge-1	precision	0.35
	recall	0.41
	f-score	0.38
Rouge-2	precision	0.33
	recall	0.36
	f-score	0.34
Rouge-L	precision	0.32
	recall	0.4
	f-score	0.38

Table 5. Example of the automatic text summarization

Manual summarization	Automatic summarization	Sentence weight (importance indicator)	Feature vectors
Биылғы мамыр-шілде айларында 2 миллион тоннадан астам көмір тасымалданды. Бұл өткен жылдың сәйкес мерзімімен салыстырғанда 35% жоғары көрсеткіш. Маусымда 850 мың тонна көмір тасымалданса, ол өткен жылдың сәйкес мерзімімен салыстырғанда 74% жоғары көрсеткішті құрады.	Жыл басынан бері 144 миллион тонна жүк тиеліп, өткен жылмен салыстырғанда 7% өсті	0.76666 - high	[0.31, 0.62, 0.5, 1.0, 1.0, 0.25, 0.25]
	Биылғы мамыр-шілде айларында 2 миллион тоннадан астам көмір тасымалданды.	0.766127 - high	[0.0, 0.77, 0.25, 0.0, 0.62, 0.1, 0.2]
	Маусымда 850 мың тонна көмір тасымалданса, ол өткен жылдың сәйкес мерзімімен салыстырғанда 74% жоғары көрсеткішті құрады	0.766127 - high	[0.0, 0.46, 0.16, 0.33, 0.45, 0.17, 0.17]

summary content of a large flow of information. The algorithm of the extractive method of abstracting using fuzzy logic has proved to be effective for the tasks of automatic summarizing of news articles dataset in the Kazakh language. In this work we presented an extractive summarization method on the basis of fuzzy logic. Our approach is advanced by applying morphological analysis and pronoun resolution techniques. The experiments conducted on Kazakh news and CNN/ Daily Mail dataset show perspective results. Nevertheless, the

algorithm requires improvements and the inclusion of additional methods in the algorithm, which will allow not only to extract important content, but also to paraphrase in order to more closely correspond to the manual abstract. As a future work, we aim to increase our dataset. Moreover, we want to convert the extracted summaries to abstractive one using neural network techniques.

## 6 Acknowledgments

This work was supported by colleagues from the University of Texas at San Antonio: Ramin Sahba, professor Mo Jamshidi. The work of Zhanibek Kozhimbayev and Zhandos Yessenbayev has been conducted under the Research grant N<sup>o</sup>AP05134272 funded by the Ministry of Education and Science of the Republic of Kazakhstan.

## References

1. Allahyari, M., Pouriyeh, S., Assefi, M., Safaei, S., Trippe, E. D., Gutierrez, J. B., & Kochut, K. (2017). Text summarization techniques: a brief survey. arXiv preprint arXiv:1707.02268.
2. Assylbekov, Z., Myrzakhmetov, B., & Makazhanov, A. (2016). Experiments with russian to kazakh sentence alignment. the 4-th International Conference on Computer Processing of Turkic Languages.
3. Barzilay, R. & Elhadad, M. (1999). Using lexical chains for text summarization. *Advances in automatic text summarization*, pp. 111–121.
4. Binwadhan, M. S., Salim, N., & Suanmali, L. (2009). Integrating of the diversity and swarm based methods for text summarization. The 5th postgraduate annual research seminar (PARS), pp. 17–19.
5. Calvo, H., Carrillo-Mendoza, P., & Gelbukh, A. (2018). On redundancy in multi-document summarization. *Journal of Intelligent & Fuzzy Systems*, Vol. 34, No. 5, pp. 3245–3255.
6. Carbonell, J. & Goldstein, J. (1998). The use of mmr, diversity-based reranking for reordering documents and producing summaries. *Proceedings of the 21st annual international ACM SIGIR conference on Research and development in information retrieval*, ACM, pp. 335–336.
7. Chen, D., Bolton, J., & Manning, C. D. (2016). A thorough examination of the cnn/daily mail reading comprehension task. arXiv preprint arXiv:1606.02858.
8. Conroy, J. M. & O'leary, D. P. (2001). Text summarization via hidden markov models. *Proceedings of the 24th annual international ACM SIGIR conference on Research and development in information retrieval*, ACM, pp. 406–407.
9. Filippova, K., Mieskes, M., Nastase, V., Ponzetto, S. P., & Strube, M. (2007). Cascaded filtering for topic-driven multi-document summarization. *Proceedings of the Document Understanding Conference*, pp. 26–27.
10. Karabalayeva, M., Yessenbayev, Z., & Kozhimbayev, Z. (2017). Regarding the impact of kazakh phonetic transcription on the performance of automatic speech recognition systems. the 5th International Conference on Turkic Languages Processing, pp. 113–129.
11. Kozhimbayev, Z., Erol, B. A., Sharipbay, A., & Jamshidi, M. (2018). Speaker recognition for robotic control via an iot device. 2018 World Automation Congress (WAC), IEEE, pp. 1–5.
12. Kozhimbayev, Z., Karabalayeva, M., & Yessenbayev, Z. (2016). Spoken term detection for kazakh language. the 4-th International Conference on Computer Processing of Turkic Languages, pp. 47–52.
13. Kozhimbayev, Z., Yessenbayev, Z., & Aibek, M. (2018). Document and word-level language identification for noisy user generated text. *IEEE 12th International Conference Application of Information and Communication Technologies*, IEEE, Almaty, Kazakhstan.
14. Kozhimbayev, Z., Yessenbayev, Z., & Karabalayeva, M. (2017). Kazakh and russian languages identification using long short-term memory recurrent neural networks. 11th IEEE International Conference on Application of Information and Communication Technologies, IEEE, pp. 342–247.
15. Lin, C.-Y. (2004). Rouge: A package for automatic evaluation of summaries. *Text Summarization Branches Out*.
16. Makazhanov, A., Myrzakhmetov, B., & Kozhimbayev, Z. (2017). On various approaches to machine translation from russian to kazakh. the 5th International Conference on Turkic Languages Processing, pp. 195–209.
17. Manning, C., Surdeanu, M., Bauer, J., Finkel, J., Bethard, S., & McClosky, D. (2014). The stanford corenlp natural language processing toolkit. *Proceedings of 52nd annual meeting of the association for computational linguistics: system demonstrations*, pp. 55–60.
18. Miranda-Jiménez, S., Gelbukh, A., & Sidorov, G. (2014). Conceptual graphs as framework for

- summarizing short texts. *International Journal of Conceptual Structures and Smart Applications (IJCSSA)*, Vol. 2, No. 2, pp. 55–75.
19. Mussina, A., Aubakirov, S., Ahmed-Zaki, D., & Trigo, P. (2019). Automatic document summarization based on statistical information. *Journal of Mathematics, Mechanics and Computer Science*, Vol. 96, No. 4, pp. 76–87.
  20. Myrzakhmetov, B. & Kozhirbayev, Z. (2018). Extended language modeling experiments for kazakh. *CEUR Workshop Proceedings*, volume 2303.
  21. Myrzakhmetov, B., Yessenbayev, Z., & Aibek, M. (2018). Initial normalization of user generated content: Case study in a multilingual setting. *IEEE 12th International Conference Application of Information and Communication Technologies*, IEEE, Almaty, Kazakhstan.
  22. Orynbayeva, A. (2017). Automatic summarization. *Proceedings of the 15th International Scientific Conference on Information Technologies and Management*, Riga, Latvia, pp. 87–88.
  23. Sahba, R., Ebadi, N., Jamshidi, M., & Rad, P. (2018). Automatic text summarization using customizable fuzzy features and attention on the context and vocabulary. *2018 World Automation Congress (WAC)*, IEEE, pp. 1–5.
  24. Suanmali, L., Salim, N., & Binwahlan, M. S. (2009). Fuzzy logic based method for improving text summarization. *arXiv preprint arXiv:0906.4690*.
  25. Warner, J. & Sexauer, J. (2017). *scikit fuzzy, twmeggs, ams, a. Unnikrishnan, G. Castelo, F. Batista, TG Badger, & H. Mishra (2017, October). Jdwarner/scikit-fuzzy: Scikit-fuzzy 0.3, Vol. 1.*
  26. Zadeh, L. A. (1975). The concept of a linguistic variable and its application to approximate reasoning. *Information sciences*, Vol. 8, No. 3, pp. 199–249.

Article received on 21/01/2019; accepted on 04/03/2019.  
Corresponding author is Altanbek Zulkhazhav.



# KeyVector: Unsupervised Keyphrase Extraction Using Weighted Topic via Semantic Relatedness

Alymzhan Toleu, Gulmira Tolegen, Rustam Mussabayev

Institute of Information and Computational Technologies, Almaty,  
Kazakhstan

{alymzhan.toleu, gulmira.tolegen.cs, rmusab}@gmail.com

**Abstract.** Keyphrase extraction is a task of automatically selecting topical phrases from a document. We present KeyVector, an unsupervised approach with weighted topics via semantic relatedness for keyphrase extraction. Our method relies on various measures of semantic relatedness of documents, topics and keyphrases in the same vector space, which allow us to compute three keyphrase ranking scores: global semantic score, find more important keyphrases for a given document by measuring the semantic relation between documents and keyphrase embeddings; topic weight, pruning/selecting the candidate keyphrases on the topic level; topic inner score, ranking the keyphrases inside each topic. Keyphrases are then generated by ranking the values of combined three scores for each candidate. We conducted experiments on three evaluation data sets of different length documents and domains. Results show that KeyVector outperforms state of the art methods on short, medium and long documents.

**Keywords.** Keyphrase extraction, clustering, topic modeling, semantic relatedness, text mining.

## 1 Introduction

Keyphrase extraction aims to automatically extract keyphrases from a document and ensure the selected keyphrases convey the main topic of the document. Key phrases are an essential component for solving the tasks of information retrieval [17, 21, 9, 19], summarization [6], text mining and topic modeling [2]. Word/phrase embeddings are distributed representations of text in an  $n$ -dimensional space. In such space, the semantic/syntactic features of words can be captured by the embeddings, and the machine

learning algorithms could reach better results in natural language processing (NLP) tasks by grouping words/phrases. Owing to its importance, the embedding becomes necessary for solving many NLP tasks [16, 22, 23] better nowadays. Many graph and topic-based approaches (TextRank [15], SingleRank [24], TopicRank [4]) for keyphrase extraction have been proposed to use internal and external discrete features such as positional features, word frequency, co-occurrences and some other Wikipedia-based statistical features. Instead of relying on either internal or external discrete features, in this paper, we present KeyVector, an unsupervised keyphrase extraction method by computing the semantic relatedness of words/phrases through embeddings.

Our approach has several advantages over existing state of the arts.

(1) Global semantic score. Embedding the sentences, candidate keyphrases into the same vector space, which allows us to compute their semantic relatedness more efficiently than discrete features. Based on the embeddings, we propose to use the global semantic score that is the semantic relatedness between document and keyphrases.

(2) Weighted topics. Ranking a large number of candidate keyphrases is often tricky. Intuitively, it is more efficient if we group the candidates into topics by their embeddings, then ranking them on the topic or global level. To do this, we compute representation for each topic after the clustering process and assign a weight for each topic by measuring the semantic relatedness between topic and documents. We show that the technique of

weighted topic influences the process of keyphrase selection.

(3) For each candidate inside each topic, we propose to compute a local ranking score, and we refer to it as topic inner semantic score. We use three standard data sets of different document size and domain to evaluate KeyVector.

We compare KeyVector to five different state of the art approaches. Experiments show that KeyVector outperforms other baselines on short/medium/long documents. It yields better results for both short and long documents. It indicates that KeyVector has better stability in its performances against the various length of documents compared to other topic-based approaches.

The rest of the paper is organized as follows: Section 2 presents the existing methods for the keyphrase extraction task; Section 3 describes the details of KeyVector; Section 4 describes the evaluation process and report the experimental results; Section 5 concludes this work.

## 2 Related Work

In general, keyphrase extraction methods can be classified into two groups: supervised and unsupervised approaches. In supervised approaches [14, 10], the problem of keyphrase extraction is regarded as a binary classification task and learn models from training data. Many details about the supervised methods and statistical features for keyphrase extraction can be found in the survey [7]. Here, we focus on unsupervised approaches that often have two ways: corpus-dependent and corpus-independent. The former requires all documents to do the extraction of keyphrases, and the TFIDF [20] is the simple, widely used approach contains two features: term-frequency and inverse document frequency. Methods belonging to the latter like TextRank [15], KeyCluster [13] TopicRank [12] and EmbedRank [1], including our proposed method, requires no other documents than the one document from which to extract keyphrases.

TextRank [15] is the well-known graph-based approach, and it builds a graph from one document. Each node of the graph corresponds

to candidate keyphrases and the edge connects two candidates.

For each node, calculate the score from other nodes connected by the edges. The top-ranked nodes from the graph are then selected as keyphrases. KeyCluster [13] is the clustering-based approach that clustering semantically similar candidates using statistical features such as word co-occurrences and positional features etc. The main idea of this method is that a candidate is to be selected as keyphrase if the candidate close to the centroid of a cluster. The clusterized candidates can be viewed as topics that a document covers. The drawback of this method is that those unimportant topics in the document could be selected as keyphrases, which is limiting the quality of the resulting sets of keyphrase. TopicRank [12] was proposed to overcome the weakness of the KeyCluster. In order to ensure that extracted keyphrases cover the main topics, this method uses Latent Dirichlet Allocation [2] to generate topics for a document and uses TextRank multiple times for a document and once for the generated topics.

EmbedRank [1] is an embedding-based approach that computes the document embedding and the embedding of each candidate phrase separately. The embeddings are obtained from the popular Doc2vec [11] and Sent2vec [18] models. The top keyphrases are selected by ranking the candidate phrases according to their cosine distance to the document embedding. EmbedRank is comparable to KeyVector, but they are different in several points: 1) the global semantic score is used to compute the semantic relation between sentences and keyphrases. 2) weighted topics are applied to do global pruning for the candidates that unlikely to be keyphrase. 3) topic inner score is used to rank the keyphrases in each topic.

## 3 KeyVector: Automatic Keyphrase Extraction

In this section, we introduce KeyVector, a novel weighted topic keyphrase extraction method via semantic relatedness, which is designed to handle the problematic situation of the task when each of



the documents has a large number of candidate keyphrases to rank.

Before describing the method, let us clarify the elements described in the following sections. Keyphrase, it is made up of one or multiple words. Candidate keyphrase, it is extracted for each document by using heuristic rules (Section 4.2) and each document has a large number of candidates. Gold Keyphrases, they are given by the annotators or authors of data sets. Topic, it consists of a set of candidate keyphrases and each document contains several topics. It also treated as cluster/group.

The method consists of three main steps: 1) project both sentences and candidate keyphrases into the same and high dimensional space to compute their semantic relatedness. 2) compute the weights to topics by clustering the candidate keyphrases of each document. 3) Obtaining the ranking scores by measuring the semantic relatedness between the candidate with the sentences, and the inner semantic score of each candidate in a topic plus topic weights.

The global architecture of KeyVector is given in Fig 1. The process of keyphrase extraction is from words ( $w$ ) to sentence ( $s$ ), then to keyphrase ( $p$ ). The edge arrows between words and sentences mean that the embeddings of each sentence are computed by averaging each word embeddings. The edge arrows between sentences and candidate mean the computation of the semantic relatedness between them.

### 3.1 Embedding the Sentences and Phrases

Representing text such as words, sentences and documents into vector representation, which allows the model to capture the semantic relatedness via word/phrase vectors within the shared high-dimensional vector space. We use this property to rank the candidate keyphrases, which allow us to partially capture the semantics between text and candidate keyphrase to meet the informativeness of keyphrase.

We represent each sentence and candidate phrase into vector representation by using word embeddings [16]. More formally, for a given collection of documents  $d \in D$ , we segment the

sentences  $S \in d$ ,  $s_i \in S$  and tokenize them into words  $W \in S$ ,  $w_l \in W$ . With the purpose of putting the method as simple as possible, we compute the sentence embedding by using the averaged vector sum of each word in the sentence.

$$\mathbf{s}_i = \frac{1}{|s_i|} \sum_{l=1}^{|s_i|} \mathbf{w}_l, \quad (1)$$

where  $|s_i|$  is the number of word in the sentence  $s_i$ . For simplicity, the notations within boldface denote vectors/matrix. The obtained sentence embedding is  $\mathbf{s}_i \in \mathbb{R}^{1 \times N}$ .  $N$  is the dimension. Note, the word embeddings are used only for the equation (1), in other cases we use keyphrase and sentence embeddings.

The process of computing the embedding for each candidate keyphrase  $p_j \in P$  is the same with sentence embedding. Each  $p_j$  consists of words  $w_l \in p_j$ . The generation of embedding for word sequences in this model is simple enough and feasible. This embedding method also allows us to embed arbitrary-length sequences of words. In order to compute both sentences and phrases embedding, we employ publicly available pre-trained word embeddings<sup>1</sup>, which allow both types of embeddings in a shared semantic vector space.

### 3.2 Weighted Topics

Given the sentences, the candidate keyphrases and their embeddings, we compute the semantic relatedness of them in order to cluster similar candidates by their meanings. The idea behind the clustering the candidates is to handle the problematic situation of keyphrase extraction when a large number of candidates extracted from each document. To demonstrate this situation comprehensively, let us consider the numbers of extracted candidates per document from three standard data sets: **Inspecc** [8], **DUC** [24] and **NUS** [17]. Figure 2 shows the distribution of the number of generated candidates and we could observe that the length of the input document as longer the document yield more candidate keyphrases.

<sup>1</sup><https://github.com/mmihaltz/word2vec-GoogleNews-vectors>

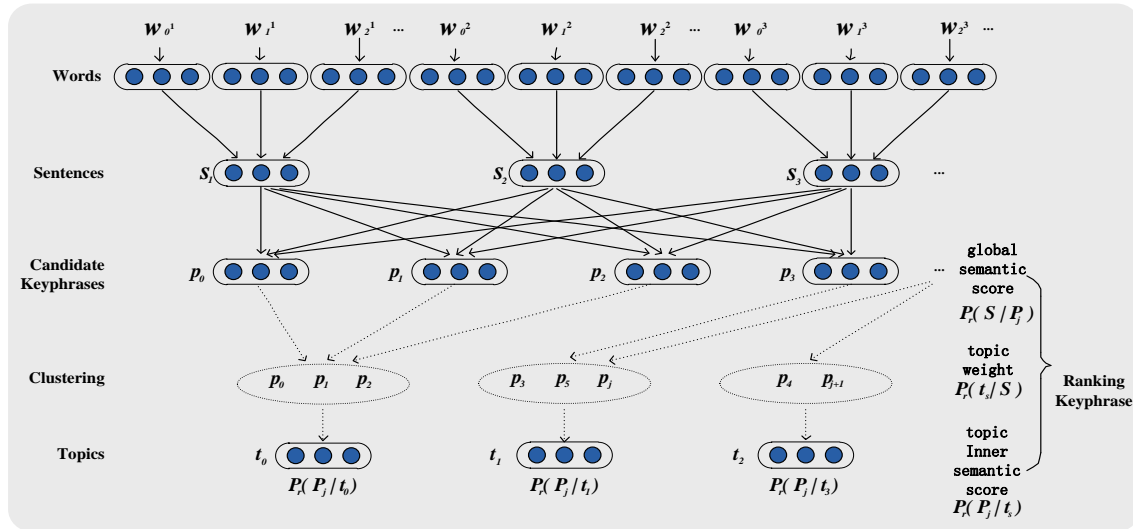


Fig. 1. The architecture of KeyVector

For NUS data set (the rightmost part of the curve), it can be seen that the extracted candidate keyphrases are more than 1000 and for DUC (the middle part of the curve) the number is about 200. The large number of candidates for each document leads the selection process become very tricky.

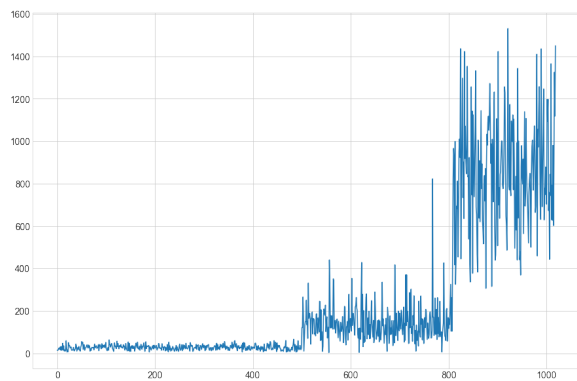


Fig. 2. The distribution of the number of generated candidates. The x-axis is the combined document ID of three data sets in the following order: Inspec (short), DUC (medium), NUS (long), and the y-axis is the number of extracted candidates

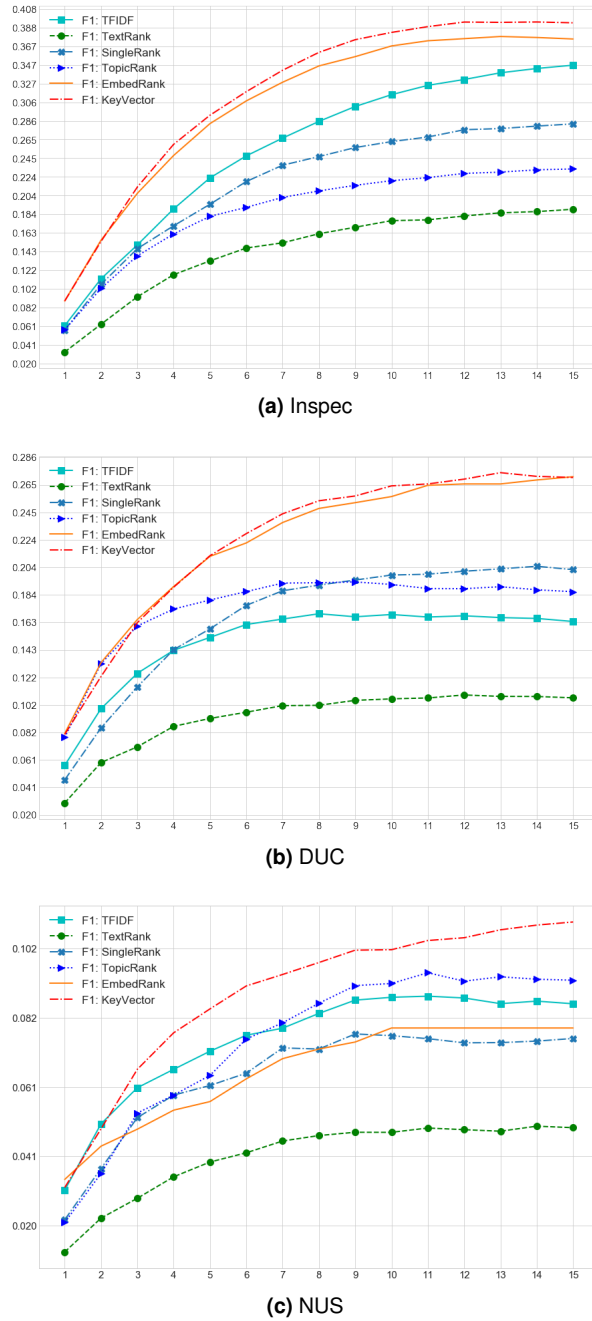
The technique of weighted topics plays the role like pruning candidates that are unlikely to be keyphrases on a global level. Existing topic-based

methods (KeyCluster [13], TopicRank [4]) apply statistical features (such as word co-occurrences, the number of overlapping words and positional features etc.) to compute the semantic relations, then group the candidates. The group of candidates can be treated as topics [4].

Here, we group the similar candidates  $P$  based on their embeddings' relatedness in the same space with the input document  $d_k$ . We apply affinity propagation method [5] to cluster the candidate in terms of a given document, and it also automatically identifies how many clusters there are in each document.

Once the candidates  $P$  are clusterized into topics,  $T = \{t_1, \dots, t_s, \dots, t_{|T|}\}$  ( $t_s$  is the  $s$ -th topic that contains one or more candidates and  $t_s$  is its embedding), again, we compute vector representation for each topic by averaging the sum of the embeddings of each candidate inside the topic. We treat topic representation  $t_s$  as the centroid of each topic. For weighting the topics, we calculate the semantic relatedness between the centroid and the input document:

$$Pr(t_s | s_i) = \sum_{i=0}^{|S_k|} (\sum_{i=0}^{|S_k|} \cos(s_i, t_s)), \quad (2)$$



**Fig. 3.** Comparison of F1-score for three data sets concerning the top- $M$  keyphrases

where  $Pr(t_s|s_i)$  is the weight for topic  $t_s$ .  $S_k \in \mathbb{R}^{|S_k| \times N}$  is the matrix representation of the

document  $d_k$  and each sentence's embedding of  $d_k$  is  $s_i \in \mathbb{R}^{1 \times N}$ .

Note that the weights for topics are computed using embeddings of sentences and topic centroids, and they are in the same vector space.

### 3.3 Keyphrase Selection

Based on the above, we select the top- $M$  keyphrases by three strategies:

(1) **global semantic score**, the semantic relatedness between sentences and candidates,  $Pr(S|p_j)$ . The underlying hypothesis of this measurement is that a sentence is more important if it contains more important keyphrases, and a candidate keyphrase is important if it is related to a large number of sentences.

(2) **topic weights**,  $Pr(t_s|S)$ . It is used for topic importance determination, and it is a global pruning technique for the candidates that are not likely to be keyphrases.

(3) **topic inner semantic score**, the semantic distances between each candidate to the centroid,  $Pr(p_j|t_s)$ . It is an inner selection in each topic to the candidates.

Given a document  $d$  or its sentences  $S$ , we compute a score for  $j$ -th candidate keyphrase by computing its likelihood:

$$Pr(p_j|S) = Pr(t_s|S)Pr(p_j|t_s)Pr(S|p_j), \quad (3)$$

where  $Pr(S|p_j) = \sum (\sum_{i=0}^{|S_k|} \cos(s_i, p_j))$  is the probability of the sentences given the keyphrase. It means that which keyphrases produce larger probabilities for sentences, it could be the keyphrases.

$Pr(p_j|t_s) = \cos(p_j, t_s)$ ,  $p_j \in t_s$  is topic inner score.

It means that the most important candidate should be close to the centroid of the topic.  $Pr(t_s|S)$  is weighted score for topic  $t_s$ . Doing the above calculation, the ranking scores for each candidate can be computed. Then, according to the ranking scores, we can suggest top- $M$  ranked candidates as the keyphrase.

## 4 Experiments

### 4.1 Data Set

KeyVector is evaluated on three publicly available data sets<sup>2</sup>. Table 1 shows the statistics about data set<sup>3</sup>.

The data set of **Inspec** [8] consists of 2 000 short documents from scientific journal abstracts. We evaluate our model on 500 documents of test set. The **DUC-2001** data set [24] contains 308 medium length newspaper articles from TREC-9. The **NUS** [17] consists of 211 long length of scientific article. Each document contains several sets of keyphrases. One is created by author and the others are assigned by annotators. Following [17], we evaluate on the union of all sets of author's and annotators' keyphrases.

### 4.2 Preprocessing

The preprocessing has impacts on the performance of keyphrase extraction models [3]. In the experiments, we used preprocessed version of Inspec [8]<sup>4</sup> and DUC-2001 [24]<sup>5</sup> data set that are publicly available. We applied following preprocessing to NUS [17] data set, namely, sentence segmentation, word tokenization and POS tagging (*nltk pos-tagger*).

Then, we extracted candidate phrases that consist of zero or more adjectives followed by one or multiple nouns. The stopwords were filtered out from the data sets and the stemming is not performed at preprocessing stage.

<sup>2</sup><https://github.com/snkim/AutomaticKeyphraseExtraction>

<sup>3</sup>The columns of Table 1 are: #docs - the number of the documents; #avg. tok. - average number of tokens per document; #avg. cand - average number of candidates; #kps - total number of keyphrases; # miss. kp. - percentage of keyphrases not present in candidates; #miss. w. - percentage of words out of vocabulary of embeddings; #miss. c. - percentage of candidates that have embedding with value 0.

<sup>4</sup><https://github.com/boudinfl/hulth-2003-pre>

<sup>5</sup><https://github.com/boudinfl/duc-2001-pre>

### 4.3 Results

To evaluate our approach, we conducted a set of experiments: one of them is to compare KeyVector with other baselines; another one is to evaluate the performance of all models concerning the number of top- $M$ .

Table 2 shows the results of KeyVector and other five baselines. Overall, KeyVector outperforms TFIDF, TextRank, SingleRank and EmbedRank in terms of precision, recall, and F1 score. On Inspec, which contains short documents, KeyVector outperforms all other competing approaches. In this case, TopicRank fails to do better than other baselines of non-topic ranked methods like TFIDF, SingleRank and EmbedRank. However, from the results of KeyVector on short documents, it indicates that the performance of KeyVector is higher than that of state of the art approaches and more stable.

On Duc, the medium documents, KeyVector also shows the case with Inspec. We could observe that KeyVector has approximately 11.95% (on Inspec), 6.62% (on Duc), 2.53% (on NUS) and 16.22% (on Inspec), 7.32% (on Duc), 0.99% (on NUS) improvements in F1-score compared with SingleRank and TopicRank, respectively

The results of KeyVector are competitive with TopicRank, and It has improvements about 0.99%, 2.53% and 5.38% compared to TopicRank, SingleRank and TextRank.

We investigate the effects of the top- $M$  selected keyphrases with respect to F1-score in Figure 3. Figure 3a shows that KeyVector outperforms all baselines start from top- $M = 3$  and it grows continuously compared to EmbedRank that start to drop slightly when top- $M=14$ . On DUC, the results of F1-score KeyVector and EmbedRank are comparable when increasing the number of top- $M$ . It can be seen that KeyVector has significant improvements compared to others. On NUS, TopicRank shows its advantages for long documents compared to others (TextRank, SingleRank) and the TFIDF also gives excellent results. KeyVector has a 0.99% improvement approximately in F1-score compare to TopicRank. KeyVector computes semantic relatedness from sentences, topics and keyphrases' embeddings

**Table 1.** Statistics of the data sets

dataset	types	#docs.	#avg. tok.	#avg. c.	#kps	%miss. kp.	%miss. w.	%miss. c.
Inspec	short	500	134.12	26.60	4913	41.66	17.5	6.78
DUC	medium	211	847.23	142.89	2488	13.46	24.76	2.74
NUS	long	308	7379.19	854.32	2317	37.16	20.13	2.21

**Table 2.** Comparison of KeyVector with state of the art on the three data sets. The  $W$  is the window size and  $M$  is the number of selected keyphrases, and the  $N$  is the dimension of word embedding

	Methods	Parameter	Precision	Recall	F1-score
Inspec	TFIDF	M=10	31.31	31.56	31.44
	TextRank	W=2, T= 0.3	17.80	17.52	17.66
	SingleRank	W=10, M=10	26.14	26.48	26.30
	TopicRank	W=10, M=10	22.00	<b>22.06</b>	22.03
	EmbedRank	N=300, M=10	36.62	36.92	36.77
	KeyVector	N=300, M=10	<b>38.09</b>	<b>38.40</b>	<b>38.25</b>
DUC	TFIDF	M=10	15.32	18.86	16.91
	TextRank	W=2, T= 0.3	9.66	11.86	10.65
	SingleRank	W=10, M=10	17.96	22.14	19.83
	TopicRank	W=10, M=10	17.34	21.34	19.13
	EmbedRank	N=300, M=10	23.23	28.64	25.65
	KeyVector	N=300, M=10	<b>23.95</b>	<b>29.52</b>	<b>26.45</b>
NUS	TFIDF	M=10	9.19	8.37	8.76
	TextRank	W=2, T= 0.3	5.02	4.57	4.78
	SingleRank	W=10, M=10	8.0	7.29	7.63
	TopicRank	W=10, M=10	9.62	8.76	<b>9.17</b>
	EmbedRank	N=300, M=10	8.24	7.50	7.86
	KeyVector	N=300, M=10	<b>10.66</b>	<b>9.71</b>	<b>10.16</b>

to do the extraction and it is sensitive for the way of generating the embeddings for sentences, topics and keyphrases including the different use of normalization/average for embeddings.

Taking into account the fact that some words are missing from the embeddings ( Inspec: 17.5%, DUC: 24.76%, NUS: 20.13% ); consequently, some of keyphrases have all zero value in representations ( Inspec: 6.78%, DUC: 2.74%, NUS: 2.21% ). We believe the improved representation for our method or decreasing the missing percentages of those words/phrases have effects on improving the results.

Over all, from the results on short/medium/long documents, we could observe that KeyVector does not fail to do better (like topicRank does) on

short/medium documents, and its performances on long documents are also stable. So we deduce that KeyVector has taken the balances on its performance when extracting keyphrases from various lengths of documents.

## 5 Conclusion

In this paper, we present KeyVector, an unsupervised method for keyphrase extraction. Our approach offers several advantages over existing keyphrase extraction methods. First, the semantic relatedness between sentences and candidates are computed through embeddings which are projected into the same high-dimensional space. The use of weighted topics captures those

unimportant topics to reach the goal of pruning the candidates not likely to be keyphrases on the topic level. The topic inner semantic score is another strategy to rank the candidate inside the topic by the semantic distances between each candidate to the topic centroid.

We conducted experiments on three standard evaluation data sets of different document sizes and domains. Results show that KeyVector outperforms other baselines on short/medium/long documents.

We will explore the following two points as future work: (1) analyze the effects of different clustering results for keyphrase extraction, and investigate other new clustering algorithms, (2) explore a new strategy for computing the topic inner scores.

## Acknowledgments

This research has been conducted within the framework of the grant num. BR05236839 “Development of information technologies and systems for stimulation of personality’s sustainable development as one of the bases of development of digital Kazakhstan”.

## References

1. **Bennani-Smires, K., Musat, C., Hossmann, A., Baeriswyl, M., & Jaggi, M. (2018).** Simple unsupervised keyphrase extraction using sentence embeddings. *CoNLL*, pp. .
2. **Blei, D. M., Ng, A. Y., & Jordan, M. I. (2003).** Latent dirichlet allocation. *J. Mach. Learn. Res.*, Vol. 3, pp. 993–1022.
3. **Boudin, F., Mougard, H., & Cram, D. (2016).** How document pre-processing affects keyphrase extraction performance. *Proceedings of the 2nd Workshop on Noisy User-generated Text (WNUT)*, The COLING 2016 Organizing Committee, Osaka, Japan, pp. 121–128.
4. **Bougouin, A., Boudin, F., & Daille, B. (2013).** Topicrank: Graph-based topic ranking for keyphrase extraction. *Proceedings of the Sixth International Joint Conference on Natural Language Processing*, Asian Federation of Natural Language Processing, pp. 543–551.
5. **Frey, B. J. & Dueck, D. (2007).** Clustering by passing messages between data points. *Science*, Vol. 315, pp. 2007.
6. **Han, J., Kim, T., & Choi, J. (2007).** Web document clustering by using automatic keyphrase extraction. *2007 IEEE/WIC/ACM International Conferences on Web Intelligence and Intelligent Agent Technology - Workshops*, pp. 56–59.
7. **Hasan, K. S. & Ng, V. (2014).** Automatic keyphrase extraction: A survey of the state of the art. *Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, Association for Computational Linguistics, Baltimore, Maryland, pp. 1262–1273.
8. **Hulth, A. (2003).** Improved automatic keyword extraction given more linguistic knowledge. *Proceedings of the 2003 Conference on Empirical Methods in Natural Language Processing*, EMNLP '03, Association for Computational Linguistics, Stroudsburg, PA, USA, pp. 216–223.
9. **Khairova, N., Petrasova, S., Lewoniewski, W., Mamyrbayev, O., & Kuralai, M. (2018).** Automatic extraction of synonymous collocation pairs from a text corpus. *Proceedings of the 2018 Federated Conference on Computer Science and Information Systems, FedCSIS 2018, Poznań, Poland, September 9-12, 2018.*, pp. 485–488.
10. **Kim, S. N. & Kan, M.-Y. (2009).** Re-examining automatic keyphrase extraction approaches in scientific articles. *Proceedings of the Workshop on Multiword Expressions: Identification, Interpretation, Disambiguation and Applications*, MWE '09, Association for Computational Linguistics, Stroudsburg, PA, USA, pp. 9–16.
11. **Lau, J. H. & Baldwin, T. (2016).** An empirical evaluation of doc2vec with practical insights into document embedding generation. *Proceedings of the 1st Workshop on Representation Learning for NLP*, Association for Computational Linguistics, pp. 78–86.
12. **Liu, Z., Huang, W., Zheng, Y., & Sun, M. (2010).** Automatic keyphrase extraction via topic decomposition. *Proceedings of the 2010 Conference on Empirical Methods in Natural Language Processing*, EMNLP '10, Association for Computational Linguistics, Stroudsburg, PA, USA, pp. 366–376.
13. **Liu, Z., Li, P., Zheng, Y., & Sun, M. (2009).** Clustering to find exemplar terms for keyphrase extraction. *Proceedings of the 2009 Conference on*

- Empirical Methods in Natural Language Processing: Volume 1 - Volume 1*, EMNLP '09, Association for Computational Linguistics, Stroudsburg, PA, USA, pp. 257–266.
14. **Lopez, P. & Romary, L. (2010).** HUMB: Automatic Key Term Extraction from Scientific Articles in GROBID. *SemEval 2010 Workshop*, ACL SigLex event, Uppsala, Sweden, pp. 4 p.
  15. **Mihalcea, R. & Tarau, P. (2004).** TextRank: Bringing order into texts. *Proceedings of EMNLP-04 and the 2004 Conference on Empirical Methods in Natural Language Processing*, pp. 404–411.
  16. **Mikolov, T., Chen, K., Corrado, G., & Dean, J. (2013).** Efficient estimation of word representations in vector space. *CoRR*, Vol. abs/1301.3781.
  17. **Nguyen, T. D. & Kan, M.-Y. (2007).** Keyphrase extraction in scientific publications. *Proceedings of the 10th International Conference on Asian Digital Libraries: Looking Back 10 Years and Forging New Frontiers*, ICADL'07, Springer-Verlag, Berlin, Heidelberg, pp. 317–326.
  18. **Pagliardini, M., Gupta, P., & Jaggi, M. (2018).** Unsupervised learning of sentence embeddings using compositional n-gram features. *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers)*, Association for Computational Linguistics, pp. 528–540.
  19. **Petrasova, S., Khairova, N., Lewoniewski, W., Mamyrbayev, O., & Mukhsina, K. (2018).** Similar text fragments extraction for identifying common wikipedia communities. *Data*, Vol. 3, No. 4.
  20. **Salton, G. & Buckley, C. (1988).** Term-weighting approaches in automatic text retrieval. *Inf. Process. Manage.*, Vol. 24, No. 5, pp. 513–523.
  21. **Tolegen, G., Toleu, A., & Zheng, X. (2016).** Named entity recognition for kazakh using conditional random fields. *Proceedings of the 4-th International Conference on Computer Processing of Turkic Languages TurkLang 2016*, Izvestija KGTU im.I.Razzakova, pp. 118–127.
  22. **Toleu, A., Tolegen, G., & Makazhanov, A. (2017).** Character-aware neural morphological disambiguation. *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, Association for Computational Linguistics, Vancouver, Canada, pp. 666–671.
  23. **Toleu, A., Tolegen, G., & Makazhanov, A. (2017).** Character-based deep learning models for token and sentence segmentation. *Conference: 5th International Conference on Turkic Languages Processing (TurkLang 2017)*, Kazan, Tatarstan, Russian Federation, pp. .
  24. **Wan, X. & Xiao, J. (2008).** Single document keyphrase extraction using neighborhood knowledge. *AAAI*, pp. 855–860.

Article received on 26/01/2019; accepted on 04/03/2019.  
Corresponding author is Aлымжан Төлеу.





# Mining Purchase Intent in Twitter

Rejwanul Haque, Mohammed Hasanuzzaman, Andy Way

Dublin City University, School of Computing,  
ADAPT Centre, Dublin,  
Ireland

{firstname.lastname}@adaptcentre.ie

**Abstract.** Most social media platforms allow users to freely express their beliefs, opinions, thoughts, and intents. Twitter is one of the most popular social media platforms where users post their intent to purchase. A purchase intent can be defined as measurement of the probability that a consumer will purchase a product or service in future. Identification of purchase intent in Twitter sphere is of utmost interest as it is one of the most long-standing and widely used measures in marketing research. In this paper, we present a supervised learning strategy to identify users' purchase intent from the language they use in Twitter. Recurrent Neural Networks (RNNs), in particular with Long Short-Term Memory (LSTM) hidden units, are powerful and increasingly popular models for text classification. They effectively encode sequences with varying length and capture long range dependencies. We present the first study to apply LSTM for purchase intent identification task. We train the LSTM network on semi-automatically created dataset. Our model achieves competent classification accuracy ( $F_1 = 83\%$ ) over a gold-standard dataset. Further, we demonstrate the efficacy of the LSTM network by comparing its performance with different classical classification algorithms taking this purchase intent identification task into account.

**Keywords.** Social media, purchase intent, mining, user generated content.

## 1 Introduction

Sharing personal thoughts, beliefs, opinions, and intents on the internet (especially on social media platforms) has become an essential part of life for millions of users all around the world. Twitter<sup>1</sup>, one of the popular social media platforms, where

users put forth their intent to purchase products or services and looks for suggestions that could assist them. To exemplify, 'I wanna buy an iPhone this week!' indicates the user's intent for buying an Apple iPhone soon. Essentially, identification and classification of such user generated contents (UGC) have twofold benefits: (a) commercial companies could exploit this to build their marketing tool/strategy, and (b) it could benefit social media users with the suggestions of the products or services that they want to purchase.

Gupta et al. [7] investigated the relationship between users' purchase intent from their social media forums such as Quora<sup>2</sup> and Yahoo! Answers<sup>3</sup>. They mainly carried out text analysis (e.g. extracting features, such as purchase action words, using the dependency structure of sentences) to detect purchase intent from UGC. In another study [18], the authors investigated the problem of identifying purchase intent. In particular, the authors (i.e. [18]) proposed a graph-based learning approach to identify intent tweets and classify them into six categories, namely 'Food & Drink', 'Travel', 'Career & Education', 'Goods & Services', 'Event & Activities' and 'Trifle'. For this, they retrieved tweets with a bootstrap method, with using a list of seed intent-indicators (e.g. 'want to'), and manually created training examples from the collected tweets. There is a potential problem in their data set since it was created based on a handful of keywords (i.e. intent-indicators).

<sup>2</sup>[www.quora.com](http://www.quora.com)

<sup>3</sup>[www.answers.yahoo.com](http://www.answers.yahoo.com)

<sup>1</sup><https://twitter.com/>

In reality, there could have many lexical variations of an intent-indicator. For example, any of these following intent-indicators can take the place of 'want to': 'like to', 'wish to', and 'need to'. Tweets often include misspelled short or long words depending on user's emotions, thoughts and state of mind.

For example, when a user is really excited to buy a car soon, his purchase intent tweet can be 'I lllllllllike to buy the SUV this month!!!' that includes an intent-indicator 'lllllllllike to' which has a misspelled long word, 'lllllllllike'.

In this work, in order to capture new tweets that are good paradigms of purchase intentions, we adopted a seed intent-indicators expansion strategy using a query expansion technique [14]. This technique has essentially helped to increase the coverage of keywords in our training data.

We manually create a labeled training data with the tweets that were extracted using a python API, given the expanded seed list of the intent-indicators. In order to identify users' purchase intention in tweets, we present a RNN model [17, 20] with LSTM units [8] (cf. Section 6). To summarize, our main contributions in this paper are as follows:

1. We are the first to apply the deep learning techniques for the users' purchase intent identification task in social media platform.
2. We create a gold-standard training data set, which, in practice, can be viewed as an ideal data set for the users' purchase intent identification task in Twitter.

The remainder of the paper is organised as follows. In Section 2, we discuss related work. Section 3 presents an existing training data that was previously used in the purchase intent identification task. In Section 4, we detail how we created training data for our experiments. In Section 5, we present our experimental methodology. Section 6 presents our experimental set-up, results and analysis, while Section 7 concludes, and provides avenues for further work.

## 2 Related Work

Identifying wishes from texts [16, 6] is apparently a new arena in natural language processing (NLP). Notably, Ramanand et al. [16] focus on identifying wishes from product reviews or customer surveys, e.g. a desire to buy a product. They primarily describe linguistic rules that can help detect these 'wishes' from text. In general, their rule-based method for identifying wishes from text proved to be effective. However, the creation of rules is a time-consuming task, and their coverage is not satisfactory. Detection of users' purchase intent in social media platform is close to the task of identifying wishes in product reviews or customer surveys.

In information retrieval (IR), query intent can broadly be classified into two categories: query type [10, 3] and user type [2, 13, 9]. The focus on this paper is to identify and classify tweets that explicitly express users' purchase intents. In this sense, this work can be kept under of the first category. To the best of our knowledge, the most relevant works to ours come from [7, 18]. In fact, to a certain extent, our proposed methods can be viewed as the extension of [18].

[7] investigated the problem of identifying purchase intent in UGC, with carrying out an analysis of the structure and content of posts and extracting features from them. They make use of the linguistic preprocessors for feature extraction, such as dependency parser and named entity recogniser, which are only available for a handful of languages.

[18] presented a graph-based learning approach to inferring intent categories for tweets. [18] primarily focus on identifying and classifying tweets that explicitly express user's purchase intentions. In order to prepare a training data with tweets that express user's purchase intentions, [18] proposed a bootstrap-based extraction model that made use of a seed list of purchase-indicators (e.g. 'want to'). They took help of manual annotators to classify the collected tweets into six different intent categories. The major disadvantage of these methods [7, 18] lies with their data set since their training data is based on a handful of keywords. In our work, we encountered this problem with employing an query

expansion technique [14], which has essentially helped to increase the coverage of keywords in our training data. We are the first to train our models with deep learning technique (RNN with LSTM hidden units) for this problem, i.e. purchase intent identification task in social media platform.

### 3 Existing Dataset

This section details an existing labeled training data in which each tweet is associated with an appropriate purchase intent or non-intent category. The creation of this data set was based on a limited set of keywords. A brief overview of this dataset is provided below.

As mentioned in Section 2, [18] applied a bootstrapping based method to retrieve intent tweets from Twitter, given a seed set of intent-indicators, (e.g. 'want to'). A manual annotation process was carried out on those extracted tweets that contain at least one intent-indicator. In short, tweets were distinguished as intent and non-intent tweets and the intent tweets were categorised into six different categories, namely 'Food and Drink', 'Travel', 'Education and Career', 'Goods and Services', 'Event and Activities' and 'Trifle'. [18] shared their training data with us. From now, we call this data set *Dataset1*. The statistics of Dataset1 can be found in [18], which we also report in Table 1. As can be seen from Table 1, Dataset1 consists of 2,263 labeled tweets, with six intent and non-intent categories.

**Table 1.** The statistics of the existing training data set, Dataset1

Category	tweets	%
Food & Drink	245	11.50%
Travel	187	8.78%
Carrer & Education	159	7.46%
Goods & Services	251	11.78%
Event & Activities	321	15.07%
Trifle	436	20.47%
Non-intent	531	24.92%
Total	2,263	

## 4 Our Dataset

This section details creation of a new training data. First, we explain why the existing data (i.e. Dataset1), to a certain extent, is inadequate for this task. Then, we demonstrate how we created a new dataset. The created dataset, in practice, is to be an ideal data set for addressing this problem.

### 4.1 Variation of Intent-Indicators

The expression of interest of a Twitter user may be associated with the user's state of mind, emotion, or other phenomenon. Hence, the different Twitter users can express their thoughts of interest in numerous ways. For example, the users may show their interest to purchase a product with any of the following intent-indicators: 'want to', 'need to', 'like to', 'wish to', and 'hope to'. Spelling mistake is a common phenomenon in tweets (e.g. short form, noisy long form). Hence, tweets may include misspelled intent-indicators.

For example, when a user is really excited to buy a product soon, his purchase intent tweet can be 'I wannnnntttt to buy an iPhone!!!!!!'. Similarly, intent indicators can be specified as 'n33d to', 'h0pe to', 'wnt to' and so on. All the prior studies in this direction do not take into the consideration of different ways by which an intent indicator can be specified. In this work, we aim to make the list of purchase intent indicators as exhaustive as possible, with taking the nature of the user generated contents in this media into consideration. In the next section we describe how we expand the existing list of seed purchase intent indicators.

### 4.2 Expanding the List of Intent-Indicators

As discussed above, the existing data set (i.e. Dataset1) has limited coverage of the intent-indicators. In order to increase the coverage, and to capture new tweets that are good paradigms of purchase intentions, we expand the list of intent-indicators<sup>4</sup> using a query expansion technique. This is accomplished with a continuous distributed vector representation of words using

<sup>4</sup>We obtained the initial list of intent-indicators from [18]

$$\frac{1}{|V|} \sum_{n=1}^{|v|} \sum_{-c \leq i \leq c, i \neq 0} \log p(w_{n+1}|w_n), \quad (1)$$

**Table 2.** Top 10 similar words/phrases of the seed intend indicators: 'want', 'need', 'wish' and 'like'

	Intent-indicators			
	want	need	wish	like
Top-10 similar words/phrases	wamt	needd	Wish	lile
	wan't	nees	wished	likr
	wanr	neeeed	wishh	llike
	want/need	neeeeed	whish	likw
	wabt	meed	Wishhhh	lik
	wNt	Need	Wished	lkke
	wnat	n99ed	Wishin	like
	/want/	neex	WISH	lije
	need/want	need/want	lwish	lke
	eant	neeeeeed	wishhh	lyk

### 4.3 Collecting Tweets with the Expanded seed Intent-Indicators

**Table 3.** A few of the tweets collected with the seed intent-indicator: 'n33d'

tweets with intent-indicator: 'n33d'

---

I n33d yall to be more active on younow plz and  
thanks

I n33d more youtubers to watch

I n33d to make some fucking music

I n33d to inhale these pizz@ pringle

I am so hungry people got to watch out n33d  
foOOoOod

I n33d to stop buying jackets

I was at ritz last friday shown out y3w n33d to go  
out to da club wit m3

I think I need to go get some therapy

We randomly sampled a set of 2,500 tweets from the list of the collected tweets that contain at least one intent-indicator, and another set of 2,500 tweets from Twitter, each of them contains no intent-indicators. During sampling we ensure that none of the tweets overlaps with those from Dataset1 (cf. Section 3). Then, we applied a noise cleaning method on the tweets, i.e. all the null values, special characters, hashtags were removed from tweets. This cleaning process was carried out with a manual editor who has excellent English skills and good knowledge on tweets. After manual cleaning, we get a set of 4,732 tweets.

<sup>5</sup><http://docs.tweepy.org/en/v3.5.0/api.html>

As mentioned earlier in the paper, [18] defined a set of purchase intent categories (six) in order to classify those tweets that express users' purchase intention. Following [18] we label each of the collected clean tweets with either one of the purchase intent categories or the non-intent category. The manual annotation process is accomplished with a GUI that randomly displays a tweet from the set of 4,732 tweets. The GUI lists the six intent (i.e. 'Food and Drink', 'Travel', 'Education and Career', 'Goods and Services', 'Event and Activities' and 'Trifle') categories and the sole non-intent category as in [18]. For the annotation purposes we hired three annotators who are native English speakers and have excellent knowledge on UGCs (i.e. tweets). The annotators are instructed to follow the following rules for labeling a tweet:

- label each tweet with an appropriate category listed on GUI,
- skip a tweet for annotation if you are unsure about user's purchase intention in the tweet or its intention category,
- skip those tweets for annotation that are unclear and includes characters of other languages or noisy characters,
- label those tweets with the non-intent category that express negative intents (e.g., 'don't want to').

On completion of the annotation task, we obtained 4,210 tweets, each of which is associated with at least one tag<sup>6</sup>. Since we have three annotators and three values are associated with the most of tweets of the set of 4,210 labeled tweets, final class for a tweet is determined with two out of three voting logic.

Thus, 635 annotated tweets were not considered in the final annotated set due to the disagreements of all three annotators. The final set of annotated tweets contains 3,575 entries. From now, we call this data set *Dataset2*, whose statistics are reported in Table 4.

<sup>6</sup>At least, one out of three manual annotators label each of the 4,210 tweets.

**Table 4.** The statistics of the new training data set, Dataset2

Category	tweets	%
Food & Drink	285	8.0%
Travel	214	6.0%
Carrer & Education	164	4.6%
Goods & Services	387	10.8%
Event & Activities	344	9.6%
Trifle	450	12.6%
Non-intent	1,803	50.4%
Total	3,575	

On completion of the annotation process, inter-annotator agreement was computed using Cohen's kappa [4] at tweet level. For each tweet we count an agreement whenever two out three annotators agree with the annotation result. We found the kappa coefficient to be very high (i.e. 0.64) for the annotation task. This indicates that our tweet labeling task is to be excellent in quality.

**Table 5.** The statistics of the combined training data set, ComDataset

Category	tweets	%
Food & Drink	530	9.1%
Travel	401	6.9%
Carrer & Education	323	5.5%
Goods & Services	538	9.2%
Event & Activities	665	11.4%
Trifle	886	15.2%
Non-intent	2,336	40.0%
Total	5,838	

#### 4.5 Combined Training Data

For our experiments we merged the training examples of Dataset1 and Dataset2. From now, we call the combined training set *ComDataset*. The statistics of ComDataset are reported in Table

5. For our experiments we randomly selected 1,000 examples from ComDataset, and create a test set with 500 examples and a validation set with 500 examples. The set of remaining 4,838 examples from ComDataset was considered as the training set.

## 5 Methodology

### 5.1 LSTM Network

Nowadays, RNN, in particular with LSTM [8] hidden units, has been proved to be an effective model for many classification tasks in NLP, e.g. sentiment analysis [19], text classification [11, 21]. RNN is an extension of the feed-forward NN, which has the gradient vanishing or exploding problems. LSTM deals with the exploding and vanishing gradient problems of RNN. An RNN composed of LSTM hidden units is often called an LSTM network. A common LSTM unit is composed of a cell, an input gate, an output gate and a forget gate. More formally, each cell in LSTM can be computed as follows:

$$X = \begin{bmatrix} h_{t-1} \\ x_t \end{bmatrix} \quad (2)$$

$$f_t = \sigma(W_f \cdot X + b_f), \quad (3)$$

$$i_t = \sigma(W_i \cdot X + b_i), \quad (4)$$

$$o_t = \sigma(W_o \cdot X + b_o), \quad (5)$$

$$c_t = f_t \odot c_{t-1} + i_t \odot \tanh(W_c \cdot X + b_c), \quad (6)$$

$$h_t = o_t \odot \tanh(c_t), \quad (7)$$

where  $W_i, W_f, W_o \in \mathbb{R}^{d \times 2d}$  are the weighted matrices and  $b_i, b_f, b_o \in \mathbb{R}^d$  are biases of LSTM, which need to be learned during training, parameterising the transformations of the input, forget and output gates, respectively.  $\sigma$  is the sigmoid function, and  $\odot$  stands for element-wise multiplication.  $x_t$  includes the inputs of LSTM cell unit. The vector of hidden layer is  $h_t$ . The final hidden vector  $h_N$  represents the whole input tweet, which is passed to *softmax* layer after linearizing it into a vector whose length is equal to the number of class labels. In our work, the set of class labels includes intent and non-intent categories.

### 5.2 Classical Supervised Classification Models

Furthermore, we compare the deep learning model with the classical classification models. We employ the following classical supervised classification techniques:

- Baseline 1: Logistic Regression (LR),
- Baseline 2: Decision Tree (DT),
- Baseline 3: Random Forest (RF),
- Baseline 4: Naïve Bayes (NB).

These classical learning models (LR, DT, RF and NB) can be viewed as the baselines in this task. Thus, we obtain a comparative overview on the performances of different supervised classification models including LSTM network. Note that we consider default set-ups of an well-known machine learning library for our baseline classifiers (cf. Section 6).

## 6 Experiments

This section details the building of different classification models. In order to build LR, DT, RF and NB classification models, we use the well-known scikit-learn machine learning library,<sup>7</sup> and performed all the experiments with default parameters set by scikit-learn. As for the representation space, each tweet was represented as a vector of word unigrams weighted by their frequency in the tweet. For building our neural network (NN) and training the model we use Lasagne library<sup>8</sup>.

Our RNN model includes LSTM units. The size of input layer of the NN is 12,000. We employ layer normalisation [1] in the model. Dropout [5] between layers is set to 0.10. The size of embedding and hidden layers are 512 and 1024. The models are trained with Adam optimizer [12], with learning-rate set to 0.0003 and reshuffling the training corpora for each epoch. We use the learning rate warm-up strategy for Adam. The validation on development set is performed using cross-entropy cost function.

**Table 6.** Accuracy of classification models (set-up 1: intent and non-intent) measured with precision, recall and  $F_1$ -score metrics

		P	R	$F_1$
LR	Intent	0.79	0.89	0.82
	Non-intent	0.88	0.73	0.80
	avg/total	0.82	0.81	0.81
DT	Intent	0.75	0.81	0.78
	Non-intent	0.80	0.74	0.77
	avg/total	0.78	0.77	0.77
RF	Intent	0.76	0.89	0.82
	Non-intent	0.87	0.73	0.79
	avg/total	0.82	0.81	0.81
NB	Intent	0.76	0.89	0.82
	Non-intent	0.88	0.73	0.80
	avg/total	0.82	0.81	0.81
RNN	Intent	0.82	0.86	0.84
	Non-intent	0.88	0.77	0.83
	avg/total	0.85	0.82	<b>0.83</b>

The RNN model is trained up-to 20 epochs, and we set mini-batches of size 32 for update.

We observe the learning curve of the classification models with the following experimental set-up:

- Set-up 1: classifying tweets into the two classes: intent and non-intent. In this case, all intent sub-classes are merged into a one single intent class (cf. Table 5).
- Set-up 2: classifying tweets into seven classes: six intent categories ('Food and Drink', 'Travel', 'Education and Career', 'Goods and Services', 'Event and Activities' and 'Trifle') and one non-intent category.
- Set-up 3 (one vs all): in this set-up we select a particular intent class, and the remaining intent sub-classes are merged into one single class. Like the first set-up (Set-up 1), this

one is a binary classification task. In order to test classifiers in this set-up, we chose the following two intent classes: 'Goods and Services' and 'Trifle'.

## 6.1 Results and Discussion

We evaluate the performance our classifiers and report the evaluation results in this section. In order to measure classifier's accuracy on the test set, we use three widely-used evaluation metrics: precision, recall and  $F_1$  measures. Note that we could not directly compare the approach of [18] with ours since the source code of their model is not freely available to use. We report the evaluation results on our gold standard test set obtained with the experimental set-up 'Set-up 1' in Table 6. Here, we draw a number of observations from the evaluation results presented in Table 6:

1. We see excellent performance with all classifiers for both intent and non-intent categories.
2. As can be seen from Table 6, the precision scores are slightly higher than the recall scores for the non-intent category. The opposite scenario is observed with the intent category. When we compare intent and non-intent categories in terms of precision and recall, we see differences of the recall scores are higher than that of the precision scores irrespective of the classification models.
3. Irrespective of the classification models, the accuracy ( $F_1$ ) of identifying purchase intent tweets is slightly better than that of identifying non-intent tweets.
4. When we compare the scores of different classification models on the test set, we see that the RNN model becomes the winner, with achieving a  $F_1$  of 0.83 on the gold-standard test set.

Next, we report the evaluation results obtained with the experimental set-up 'Set-up 2' (cf. Section 6) in Table 7 and 8. This time, the classification task involves seven output classes, i.e. six intent classes and the sole non-intent class.

<sup>7</sup><https://scikit-learn.org/stable/>

<sup>8</sup><https://lasagne.readthedocs.io/en/latest/>

**Table 7.** Accuracy of the LR, DT and RF models (set-up 2) on six intent classes and one non-intent class measured with precision, recall and  $F_1$ -score metrics.

		P	R	$F_1$
LR	Food and Drink	0.87	0.83	0.85
	Travel	0.69	0.57	0.62
	Education & Career	1.0	0.51	0.68
	Goods & Services	0.77	0.56	0.65
	Event & Activities	0.71	0.51	0.68
	Trifle	0.47	0.45	0.46
	Non-intent	0.78	0.94	0.86
	avg/total	0.75	0.75	0.74
DT	Food and Drink	0.54	0.42	0.30
	Travel	0.34	0.37	0.36
	Education & Career	0.54	0.54	0.54
	Goods & Services	0.60	0.57	0.59
	Event & Activities	0.29	0.31	0.30
	Trifle	0.35	0.47	0.40
	Non-intent	0.80	0.76	0.78
	avg/total	0.63	0.61	0.62
RF	Food and Drink	0.61	0.69	0.65
	Travel	0.49	0.54	0.51
	Education & Career	0.71	0.57	0.63
	Goods & Services	0.62	0.71	0.66
	Event & Activities	0.54	0.28	0.37
	Trifle	0.35	0.34	0.34
	Non-intent	0.80	0.84	0.82
	avg/total	0.67	0.68	0.67

**Table 8.** Accuracy of the NB and RNN models (set-up 2) on six intent classes and one non-intent class measured with precision, recall and  $F_1$ -score metrics.

		P	R	$F_1$
NB	Food & Drink	1.0	0.12	0.22
	Travel	1.0	0.03	0.06
	Education & Career	1.0	0.06	0.11
	Goods & Services	0.89	0.18	0.30
	Event & Activities	1.0	0.03	0.05
	Trifle	0.43	0.04	0.07
	Non-intent	0.54	1.00	0.70
	avg/total	0.69	0.55	0.43
RRN	Food & Drink	0.90	0.85	0.88
	Travel	0.76	0.68	0.72
	Education & Career	0.74	0.73	0.74
	Goods & Services	0.86	0.67	0.75
	Event & Activities	0.92	0.96	0.94
	Trifle	0.57	0.54	0.55
	Non-intent	0.67	0.91	0.77
	avg/total	0.77	0.76	<b>0.76</b>

Note that due to the space constraints, we report the results in two tables (i.e. Tables 7 and 8). Here, we draw a number of observations from the evaluation results presented in Table 7 and 8:

1. In general, we get high precision and low recall scores for the intent categories. For the non-intent class, most of the cases, as in above, the scenario is the other way round.
2. As far as the scores obtained with the  $F_1$  metric are concerned, we see that the RNN and LR classifiers performed reasonably, and the remaining classifiers (i.e. DT, NB and RF) performed moderately.
3. When we compare different classification models, we see, as in Set-up 1, the RNN model becomes the winner, with achieving  $F_1$  of 0.76 (average) on the gold-standard test set. When we see  $F_1$  scores for the intent and non-intent classes, we see that



the RNN classifiers performs consistently and outperforms all classical classification models in most of the cases.

4. As can be seen from Table 8, the recall scores of NB classifier are below par, and even in some cases, those are very poor. We recall Table 5 where we can see the presence of class imbalance in the training data. For instance, 6.9% and 5.5% training examples belong to 'Travel' and 'Career and Education' classes, respectively. This could be one of the reasons why classifiers performed poorly for some categories. This phenomenon is also corroborated by Gupta et al. [7] who built classifiers with a training data having the class imbalance issues.

Now, we observe the learning curve of the classifiers with the third experimental set-up (i.e. 'Set-up 3', cf. Section 6) where a particular intent category (e.g. 'Goods and Services' or 'Trifle') is held and the rest of the intent categories are merged into a single category. In this set-up, we remove those examples from the test and development sets that include the non-intent target class. Then, we test our classifiers on the resulting test set and obtain the evaluation results, which are reported in Table 9 ('Goods and Services') and in Table 10 ('Trifle'). Here, we draw a number of observations from the evaluation results presented in Table 9 and 10:

1. We see an excellent performance across the classifiers for 'Goods and Services' and the combined intent category.
2. We get a mix bag of results across the classifiers and metrics for 'Trifle' and the combined intent category .
3. Like the above experimental set-ups, in this set-up, the RNN models proved to be superior than the other classical classification models in identifying users' purchase intent type in tweets. The RNN model produces an accuracy of  $F_1$  of 0.95 (average) on the test set when 'Goods and Services' category is considered. As far as the 'Trifle' category is concerned, the RNN model gives an accuracy of  $F_1$  (average) of 0.83 on the test set.

**Table 9.** Accuracy of classification models (set-up 3: 'Goods and Services' and a combined class from the rest of the intent categories measured with precision, recall and F1 metrics

		P	R	$F_1$
LR	Goods & Services	0.88	0.76	0.82
	Intent	0.90	0.96	0.93
	avg/total	0.90	0.90	0.90
DT	Goods & Services	0.80	0.76	0.78
	Intent	0.90	0.92	0.91
	avg/total	0.87	0.87	0.87
RF	Goods & Services	0.80	0.72	0.76
	Intent	0.89	0.92	0.91
	avg/total	0.86	0.86	0.86
NB	Goods & Services	0.88	0.47	0.61
	Intent	0.81	0.97	0.89
	avg/total	0.83	0.82	0.80
RNN	Goods & Services	0.92	0.98	0.95
	Intent	0.94	0.98	0.96
	avg/total	0.93	0.98	<b>0.95</b>

4. When we consider the recall scores in Table 10, we see most of the classifiers performed below par with the 'Trifle' category. This anomaly needs to be investigated and we keep this topic as a subject of future work.

## 7 Conclusion

In this paper, we presented supervised learning models to identify users' purchase intent from the tweet data. We present the first study to apply LSTM network for purchase intent identification task. With our RNN classifiers we achieved

**Table 10.** Accuracy of classification models (set-up 3: 'Trifle' and a combined class from the rest of the intent categories measured with precision, recall and F1 metrics

		P	R	$F_1$
LR	Trifle	0.70	0.43	0.54
	Intent	0.83	0.94	0.88
	Avg/total	0.80	0.81	0.79
DT	Trifle	0.53	0.45	0.49
	Intent	0.82	0.86	0.84
	avg/total	0.75	0.76	0.75
RF	Trifle	0.66	0.38	0.48
	Intent	0.81	0.93	0.87
	avg/total	0.77	0.79	0.77
NB	Trifle	1.00	0.09	0.17
	Intent	0.76	1.00	0.87
	avg/total	0.82	0.77	0.69
RNN	Trifle	0.88	0.69	0.77
	Intent	0.93	0.87	0.89
	avg/total	0.91	0.78	<b>0.83</b>

competent accuracy ( $F_1$  ranging from 0.76 to 0.95) in all classification tasks. This shows applicability of the deep learning algorithms to a classification task where a tiny training data is available.

Further, we demonstrated the efficacy of the LSTM network by comparing its performance with different classifiers. The major portion of the paper describes the way we created our own training data. The existing training data set for this task was not satisfactory as it is limited with a set of keywords. We semi-automatically created training data set, with employing a state-of-the-art query expansion technique [14]. This has essentially helped to increase the coverage of keywords in our training data.

In future, we intend to make our gold standard data set available to the NLP community. We also plan to test our method on different social media platform, e.g. Facebook,<sup>9</sup> and with different

<sup>9</sup><https://www.facebook.com/>

languages. We also intent to apply our methods to a cross-lingual social platform. We plan to increase the size of training examples for those classes for which we have lesser proportion of training examples. This could encounter the class imbalance problem in our training data.

## Acknowledgements

The ADAPT Centre for Digital Content Technology is funded under the Science Foundation Ireland (SFI) Research Centres Programme (Grant No. 13/RC/2106) and is co-funded under the European Regional Development Fund. This project has partially received funding from the European Union's Horizon 2020 research and innovation programme under Marie Skłodowska-Curie Grant Agreement No. 713567, and the publication has emanated from research supported in part by a research grant from SFI under Grant Number 13/RC/2077.

## References

1. Ba, J. L., Kiros, J. R., & Hinton, G. E. (2016). Layer normalization. *CoRR*, Vol. abs/1607.06450.
2. Beitzel, S. M., Jensen, E. C., Lewis, D. D., Chowdhury, A., & Frieder, O. (2007). Automatic classification of web queries using very large unlabeled query logs. *ACM Transactions on Information Systems (TOIS)*, Vol. 25, No. 2, pp. 9.
3. Cao, H., Hu, D. H., Shen, D., Jiang, D., Sun, J.-T., Chen, E., & Yang, Q. (2009). Context-aware query classification. *Proceedings of the 32nd international ACM SIGIR conference on Research and development in information retrieval*, ACM, Boston, MA, pp. 3–10.
4. Cohen, J. (1960). A coefficient of agreement for nominal scales. *Educational and Psychological Measurement*, Vol. 20, No. 1, pp. 37–46.
5. Gal, Y. & Ghahramani, Z. (2016). A theoretically grounded application of dropout in recurrent neural networks. *CoRR*, Vol. abs/1512.05287.
6. Goldberg, A. B., Fillmore, N., Andrzejewski, D., Xu, Z., Gibson, B., & Zhu, X. (2009). May all your wishes come true: A study of wishes and how to recognize them. *Proceedings of HLT-NAACL: Human Language Technologies: The 2009 Annual*

- Conference of the North American Chapter of the Association for Computational Linguistics*, Boulder, CO, pp. 263–271.
7. Gupta, V., Varshney, D., Jhamtani, H., Kedia, D., & Karwa, S. (2014). Identifying purchase intent from social posts. *Proceedings of the Eighth International AAAI Conference on Weblogs and Social Media*, Ann Arbor, Michigan, pp. 180–186.
  8. Hochreiter, S. & Schmidhuber, J. (1997). Long short-term memory. *Neural computation*, Vol. 9, No. 8, pp. 1735–1780.
  9. Hu, J., Wang, G., Lochovsky, F., Sun, J.-t., & Chen, Z. (2009). Understanding user's query intent with wikipedia. *Proceedings of the 18th international conference on World wide web*, ACM, Madrid, Spain, pp. 471–480.
  10. Jansen, B. J., Booth, D. L., & Spink, A. (2008). Determining the informational, navigational, and transactional intent of web queries. *Information Processing & Management*, Vol. 44, No. 3, pp. 1251–1266.
  11. Joulin, A., Grave, E., Bojanowski, P., & Mikolov, T. (2016). Bag of tricks for efficient text classification. *arXiv preprint arXiv:1607.01759*.
  12. Kingma, D. P. & Ba, J. (2014). Adam: A method for stochastic optimization. *CoRR*, Vol. abs/1412.6980.
  13. Li, X., Wang, Y.-Y., & Acero, A. (2008). Learning query intent from regularized click graphs. *Proceedings of the 31st annual international ACM SIGIR conference on Research and development in information retrieval*, ACM, New York, NY, pp. 339–346.
  14. Mikolov, T., Sutskever, I., Chen, K., Corrado, G. S., & Dean, J. (2013). Distributed representations of words and phrases and their compositionality. *Advances in neural information processing systems*, pp. 3111–3119.
  15. Pennington, J., Socher, R., & Manning, C. (2014). Glove: Global vectors for word representation. *Proceedings of the 2014 conference on empirical methods in natural language processing (EMNLP)*, Doha, Qatar, pp. 1532–1543.
  16. Ramanand, J., Bhavsar, K., & Pedanekar, N. (2010). Wishful thinking: finding suggestions and 'buy' wishes from product reviews. *Proceedings of the NAACL HLT 2010 workshop on computational approaches to analysis and generation of emotion in text*, Los Angeles, CA, pp. 54–61.
  17. Rumelhart, D. E., Hinton, G. E., & Williams, R. J. (1986). Learning representations by back-propagating errors. *Nature*, Vol. 323, No. 6088, pp. 533.
  18. Wang, J., Cong, G., Zhao, W. X., & Li, X. (2015). Mining user intents in twitter: A semi-supervised approach to inferring intent categories for tweets. *Twenty-Ninth AAAI Conference on Artificial Intelligence*, Austin, TX, pp. 318–324.
  19. Wang, Y., Huang, M., Zhao, L., et al. (2016). Attention-based LSTM for aspect-level sentiment classification. *Proceedings of the 2016 conference on empirical methods in natural language processing*, Austin, TX, pp. 606–615.
  20. Werbos, P. J. (1990). Backpropagation through time: what it does and how to do it. *Proceedings of the IEEE*, Vol. 78, No. 10, pp. 1550–1560.
  21. Zhou, P., Qi, Z., Zheng, S., Xu, J., Bao, H., & Xu, B. (2016). Text classification improved by integrating bidirectional lstm with two-dimensional max pooling. *arXiv preprint arXiv:1611.06639*.

Article received on 21/01/2019; accepted on 17/02/2019.  
Corresponding author is Rejwanul Haque.



# Multi-Head Multi-Layer Attention to Deep Language Representations for Grammatical Error Detection

Masahiro Kaneko, Mamoru Komachi

Tokyo Metropolitan University,  
Graduate School of Systems Design, Tokyo,  
Japan

kaneko-masahiro@ed.tmu.ac.jp, komachi@tmu.ac.jp

**Abstract.** It is known that a deep neural network model pre-trained with large-scale data greatly improves the accuracy of various tasks, especially when there are resource constraints. However, the information needed to solve a given task can vary, and simply using the output of the final layer is not necessarily sufficient. Moreover, to our knowledge, exploiting large language representation models to detect grammatical errors has not yet been studied. In this work, we investigate the effect of utilizing information not only from the final layer but also from intermediate layers of a pre-trained language representation model to detect grammatical errors. We propose a multi-head multi-layer attention model that determines the appropriate layers in Bidirectional Encoder Representation from Transformers (BERT). The proposed method achieved the best scores on three datasets for grammatical error detection tasks, outperforming the current state-of-the-art method by 6.0 points on FCE, 8.2 points on CoNLL14, and 12.2 points on JFLEG in terms of  $F_{0.5}$ . We also demonstrate that by using multi-head multi-layer attention, our model can exploit a broader range of information for each token in a sentence than a model that uses only the final layer's information.

**Keywords.** Multi-head multi-layer attention, grammatical error detection.

## 1 Introduction

Neural networks are known to be best exploited when trained on large-scale data. It has been demonstrated that utilizing language representation models pre-trained with large-scale data is effective for various tasks. For example, recent studies have shown a significant improvement

using large-scale data to train large deeper models for natural language understanding tasks [1, 4, 11].

In contrast, for grammatical error detection, several studies have adapted large-scale data by creating artificial training data from a large-scale raw corpora [6, 14]. Moreover, there have been studies that have effectively used language representation models for grammatical error detection task [13]. To our knowledge, however, there are no studies that have utilized deep language representation models pre-trained with large-scale data for this task.

Moreover, deep neural networks learn different representations for each layer. For example, Belinkov et al. [3] demonstrated that in a machine translation task, the lower layers of the network learn to represent the word structure, while higher layers are more focused on word meaning.

Peters et al. [11] showed that in learning deep contextualized word representations, constructing representations of layers corresponding to each task by a weighted sum improved the accuracy of six NLP tasks. Peters et al. [12] empirically showed that lower layers are best-suited for local syntactic relationships, that higher layers better model longer-range relationships, and that the top-most layers specialize at the language modeling.

For tasks that emphasize the grammatical nature, such as grammatical error detection, information from the lower layers is considered to be important alongside more expressive information in deep layers. Therefore, we

hypothesized that using information from optimal layers suitable for a given task is important.

As such, our motivation is to construct a deep grammatical error detection model that considers optimal information from each layer. Therefore, we propose a model that uses multi-head multi-layer attention in order to construct hidden representations from different layers suitable for grammatical error detection.

Our contributions are as follows:

1. We propose a multi-head multi-layer attention model that can acquire even more suitable representations for a given task by fine-tuning a pre-trained deep language representation model with large-scale data for grammatical error detection.
2. We show that our model is effective at acquiring hidden representations from various layers for grammatical error detection. Our analysis reveals that using multi-head multi-layer attention effectively utilizes information from various layers. We also demonstrate that our proposed model can use a wider range of information for each token in a sentence.
3. Experimental results show that our multi-head multi-layer attention model achieves state-of-the-art results on three grammatical error detection datasets (viz., FCE, CoNLL14, and JFLEG).

## 2 Related Works

### 2.1 Grammatical Error Detection with Language Representations

Often, in sequence labeling tasks, recent supervised neural grammatical error detection models are built upon Bi-LSTM [5, 6, 13, 14, 15, 16]. Rei and Søgaard [15] used token-level predictions by Bi-LSTM for self-attention to predict sentence-level labels for grammatical error detection. However, we adopt a transformer block-based model for token-level grammatical error detection, and we build a very deep model for this task.

Rei [13] showed the effectiveness of multitask learning by coupling language modeling and grammatical error detection.

They used an additional objective for language modeling training to learn to predict surrounding tokens for every token in a dataset. In contrast to previous research, we adopt information from deep language representations for grammatical error detection by multi-head multi-layer attention.

Several studies have exploited large quantities of raw data to create additional artificial data. Rei et al. [14] artificially generated writing errors in order to create additional resources to learn a neural sequence labeling model following Rei [13]. Kasewa et al. [6] employed a neural machine translation system to create error-filled artificial data for grammatical error detection. By contrast, we directly adopt a pre-trained language representation model trained with large-scale raw data.

### 2.2 Using the Layer Representations

Deep Contextualized Word Representations (ELMo) [11] used large-scale data for a deep language representation model. Their model learns task-specific weighting from all fixed hidden layers of the pre-trained bidirectional long short-term memory (Bi-LSTM) to construct contextualized word embeddings optimized to a given task. In other words, ELMo learns task-specific representations exclusively in the first layer, whereas other parameters of a pre-trained model remain unchanged. On the contrary, we construct representations suited for given tasks by fine-tuning all parameters of our pre-trained model, using multi-head multi-layer attention. All parameters and constructed representations of our model are trained to be best-suited for the given task.

Takase et al. [18] employed intermediate layer representations, including input embeddings, to calculate the probability distributions in order to solve a ranking problem in language generation tasks. Similarly, we considered the information of each layer, but our motivation is to seize the optimal information from each layer suitable for a given task using a multi-head multi-layer attention.

Moreover, their model estimated probability distributions from each layer, whereas ours constructs hidden representations from each layer for the output layer.

Furthermore, there is a study that predicts information from the middle layer of each layer of the language model and learns the errors occurring owing to the model [2]. The use of the information of the middle layer of `transformer_block` is common to our research, but the information of each layer is not taken into account at the time of evaluation and is used only for learning. Furthermore, the information on the surface layer is less useful and learning is undertaken so that the influence of the surface layer decreases as learning progresses. In contrast, as the method uses attention, it also lets you learn which layer is utilized in the model itself.

### 3 Deep Language Representations for Grammatical Error Detection

We propose a model that applies multi-head attention to each layer (multi-head multi-layer attention, MHMLA) to fine-tune pre-trained Bidirectional Encoder Representations from Transformers (BERT) [4]. Architectures of BERT and MHMLA for the grammatical error detection task are illustrated in Figure 1. In this section, we first introduce BERT and then explain our proposed model, MHMLA.

#### 3.1 BERT

BERT is designed to learn deep bidirectional representations by jointly conditioning both the left and right contexts in all layers (Figure 1(a)). It is based on a multi-layer bidirectional transformer encoder [20]. Insofar it is a deep language representation model pre-trained on large-scale data, it can be used for fine-tuning. It achieved state-of-the-art results for a wide range of tasks such as natural language understanding, name entity recognition, question answering, and grounded commonsense inference [4].

BERT has a multi-layer bidirectional transformer encoder and can be used for different architectures, such as in classification and sequence-to-sequence learning tasks. Here, we explain the

BERT's architecture for sequence labeling tasks. Given a sequence  $S = w_0, \dots, w_n, \dots, w_N$  as input, BERT is formulated as follows:

$$h_n^0 = W_e w_n + W_p, \quad (1)$$

$$h_n^l = \text{transformer\_block}(h_n^{l-1}), \quad (2)$$

$$y_n^{(\text{BERT})} = \text{softmax}(W_o h_n^L + b_o), \quad (3)$$

where  $w_n$  is a current token, and  $N$  denotes the sequence length. Equation 1 thus creates an input embedding. Here, `transformer_block` includes self-attention and fully connected layers [20], and outputs  $h_n^l$ .  $l$  is the number of the current layer,  $l \geq 1$ .  $L$  is the total number of layers of BERT. Equation 3 denotes the output layer.  $W_o$  is an output weight matrix,  $b_o$  is a bias for the output layer, and  $y_n^{(\text{BERT})}$  is a prediction.

The parameters  $W_e$ ,  $W_p$  and `transformer_block` are pre-trained on a large document-level corpus using a masked language model [19] and predicting a next sentence. Then, BERT uses a different task-specific matrix  $W_o$  of the output layer (Equation 3) for a given sequence labeling task. To adapt BERT for specific tasks, all parameters of BERT are fine-tuned jointly by predicting a task-specific label with the task-specific output layer to maximize the log-probability of the correct label.

#### 3.2 Multi-Head Multi-Layer Attention to Acquire Task-Specific Representations

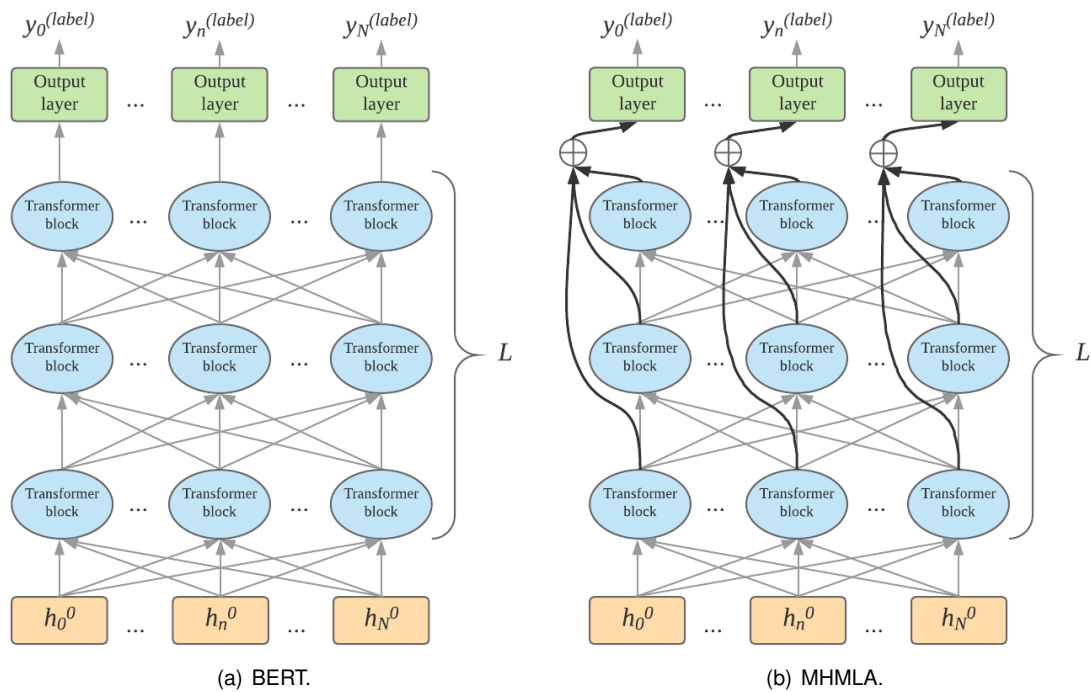
Multi-head attention [20] is more beneficial than a single attention function. MHMLA on a sequence labeling model applies attention to each layer  $l$  of the output of `transformer_block`  $h_n^l$  of Equation 2 (Figure 1(b)). First, we calculate attention value  $v_n^l$ :

$$v_{n,j}^l = W_v^l h_n^l + b_{v,j}^l. \quad (4)$$

Here,  $W_v$  is a weight matrix,  $b_v$  is a bias, and  $j$  is a head number. We apply a non-linear layer to  $h_n^l$  to acquire  $k_n^l$ . Attention score  $a_n^l$  is as follows:

$$k_{n,j}^l = \text{relu}(W_{kj}^l h_n^l + b_{kj}^l), \quad (5)$$

$$a_{n,j}^l = W_{aj}^l k_n^l + b_{aj}^l, \quad (6)$$



**Fig. 1.** Architectures of BERT and MHMLA for grammatical error detection

where  $W_k$  and  $W_a$  are weight matrices, and  $b_k$  and  $b_a$  are biases. Multi-heads are then calculated as follows:

$$\tilde{a}_{n,j}^l = \frac{\exp(a_{n,j}^l)}{\sum_{t=1}^L \exp(a_{n,j}^t)}, \quad (7)$$

$$\text{head}_{n,j} = \sum_{t=1}^L \tilde{a}_{n,j}^t v_{n,j}^t, \quad (8)$$

where  $\tilde{a}^l$  is the attention weight, normalized to sum up to 1 over all values in the layers. These weights are then used to combine the context-conditioned hidden representations from Equation (5) into a single-token representation  $c_n$ :

$$c_n = \text{concat}(\text{head}_{n,1}, \dots, \text{head}_{n,J}), \quad (9)$$

where  $J$  is the total number of heads. Finally, we return task-specific predictions based on this representation:

$$y_n^{(\text{label})} = \text{softmax}(W_o c_n + b_o). \quad (10)$$

**Table 1.** Sentence statistics of used corpora

corpus	train	dev	test
FCE	28,731	2,222	2,720
CoNLL14	-	-	1,312
JFLEG	-	-	747

$W_o$  is an output weight matrix and  $b_o$  is a bias of output layer. Our model is optimized by minimizing cross-entropy loss on the token-level annotation.

## 4 Experiments

### 4.1 Datasets

We focus on a supervised sequence labeling task: viz., grammatical error detection. Grammatical error detection is the task of identifying incorrect tokens that need to be edited in order to produce a grammatically correct sentence. We evaluated our approach on the three different grammatical



error detection datasets. Table 1 shows statistics for each corpus.

**FCE.** We fine-tuned and searched the parameters of the model and evaluated our system on the First Certificate in English (FCE) dataset [22], which contains error-annotated short essays written by language learners. The FCE dataset is a popular English learner corpus for grammatical error detection. We followed the official split of the data.

**CoNLL14.** We additionally used dataset from the CoNLL 2014 shared task (CoNLL14) dataset [10] in our evaluation. This dataset was written by higher-proficiency learners on different technical topics. It was manually corrected by two separate annotators, and we report results on each of these annotations (CoNLL14- $\{1,2\}$ ).

**JFLEG.** We also evaluated our approach with the JHU FLuency-Extended GUG (JFLEG) corpus [9]. It contains a broad range of language-proficiency levels and focuses more on fluency edits and making the text more native-sounding, in addition to grammatical corrections. JFLEG is not labeled for grammatical error detection. Therefore, we used dynamic programming to label tokens in sentences as correct or incorrect. Because JFLEG is a recently developed corpus, there is only one prior study with experimental results [15]. JFLEG is tagged by multiple annotators, like CoNLL14, so we followed this work to build a version that combines the references: if a token is labeled as an error by any annotator, it is marked as an error<sup>1</sup>.

## 4.2 Experimental Details

We used a publicly available pre-trained deep language representation model, namely the BERT<sub>BASE</sub> uncased model<sup>2</sup>. This model has 12 layers, 768 hidden size, and 16 heads of

self-attention. Layer attention has 12 heads ( $J = 12$ ). We fine-tuned the model over 5 epochs with a batch size of 32. The maximum training sentence length was 128 tokens. We used the Adam optimizer [7] with a learning rate of 5e-05. We applied dropout [17] to  $h_n^l$ ,  $k_{n,j}^l$ , and  $\tilde{a}_{n,j}^l$  with a dropout rate of 0.3.  $\tilde{a}_{n,j}^l$  is referred to as attention dropout. We also used WordPiece embeddings [21]. To make this compatible with sub-token tokenization, we inputted each tokenized word into the WordPiece tokenizer and used the hidden state corresponding to the first sub-token as input to the output layer, as with the original BERT.

We used  $F_{0.5}$  as the main evaluation measure. This measure was also adopted in the CoNLL14 shared task for the grammatical error correction task [10]. It combines both precision and recall, while assigning twice as much weight to precision, because accurate feedback is often more important than coverage in error detection applications [8].

## 4.3 Baselines and Comparisons

We compare with models of Rei [13], Rei and Søgaard [15], Rei et al. [14], and Kasewa et al. [6] which are based on the Bi-LSTM architecture. The first group, Rei [13] and Rei and Søgaard [15], was trained exclusively on the FCE dataset. The second group, Rei et al. [14] and Kasewa et al. [6] used additional artificial data along with the FCE dataset for training.

Our baseline and proposed models were trained on the transformer architecture. The first three are the descriptions of our baselines, and the fourth is a description of the proposed model:

**BERT<sub>BASE</sub> w/o pre-train.** This model is trained using only the FCE dataset and with random initialization. This baseline did not use any other corpus for training.

**BERT<sub>BASE</sub>.** This is the original pre-trained model described in Section 4.2 fine-tuned on the FCE dataset. This baseline uses original BERT model [4] and can be seen as surrogated version of the proposed method without multi-layer attention.

<sup>1</sup>Although JFLEG's experimental settings are not described in the paper, we confirmed them with the authors of the paper over e-mail.

<sup>2</sup><https://github.com/google-research/bert>

**Table 2.** Results of grammatical error detection. These results are averaged over five runs. \* and † indicate that there is a significant difference against BERT<sub>BASE</sub> and AvgL, respectively

	FCE			CoNLL14-1			CoNLL14-2			JFLEG		
	P	R	F <sub>0.5</sub>	P	R	F <sub>0.5</sub>	P	R	F <sub>0.5</sub>	P	R	F <sub>0.5</sub>
Rei [13]	58.88	28.92	48.48	17.68	19.07	17.86	25.22	19.25	23.62	-	-	-
Rei and Søgaard [15]	65.53	28.61	52.07	25.14	15.22	22.14	37.72	16.19	29.65	72.53	25.04	52.52
Rei et al. [14]	60.67	28.08	49.11	23.28	18.01	21.87	35.28	19.42	30.13	-	-	-
Kasewa et al. [6]	-	-	55.6	-	-	28.3	-	-	35.5	-	-	-
BERT <sub>BASE</sub> w/o pre-train	48.85	11.30	29.34	11.45	7.80	10.47	18.24	9.31	15.30	58.85	13.22	34.81
BERT <sub>BASE</sub>	<b>69.80</b>	37.37	59.47	34.08	<b>33.56</b>	33.97	46.01	33.89	42.93	<b>78.06</b>	36.28	63.45
AvgL	68.09	41.14	60.20	34.97	32.02	34.33	45.33	35.27	42.88	77.35	37.05	63.52
MHMLA	68.87 <sup>†</sup>	<b>43.45</b> <sup>*†</sup>	<b>61.65</b> <sup>*†</sup>	<b>35.74</b> <sup>*</sup>	33.50 <sup>†</sup>	<b>35.26</b> <sup>*†</sup>	<b>46.45</b> <sup>†</sup>	<b>35.47</b> <sup>*</sup>	<b>43.74</b> <sup>†</sup>	77.74	<b>38.85</b> <sup>*†</sup>	<b>64.77</b> <sup>*†</sup>

**Table 3.** F<sub>0.5</sub> scores of MHMLA using different number of heads  $J$ . These results are averaged over five runs

$J$	FCE	CoNLL14-1	CoNLL14-2	JFLEG
1	61.16	33.75	42.89	63.98
2	61.62	33.44	42.42	63.72
3	<b>61.90</b>	34.50	43.17	64.45
4	61.55	33.74	42.80	64.37
6	61.22	34.26	43.29	64.48
8	61.27	34.72	43.02	64.10
12	61.65	<b>35.26</b>	<b>43.74</b>	<b>64.77</b>

**AvgL.** This model is called averaged layers, which averages representations after linear transformation of  $h_n^l$  (Equation 2) for the output layer of BERT<sub>BASE</sub> model instead of using an attention.

**MHMLA.** This is the proposed model that is an extension of BERT<sub>BASE</sub>, with MHMLA to the pre-trained model while fine-tuning on the FCE dataset.

## 5 Results

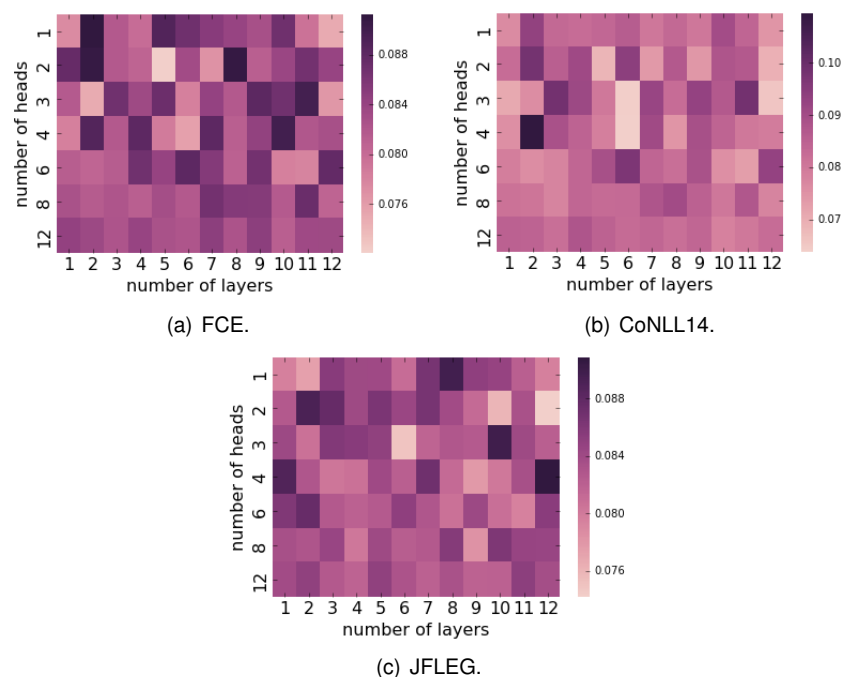
Table 2 shows the grammatical error detection results for the FCE, CoNLL14- $\{1,2\}$ , and JFLEG datasets. Scores for Rei [13], Rei and Søgaard [15], Rei et al. [14], and Kasewa et al. [6] were taken from their respective papers. In FCE, CoNLL14, and JFLEG, the BERT<sub>BASE</sub> model significantly outperformed existing methods and our baseline (without pre-training) in terms of precision, recall, and F<sub>0.5</sub>. This demonstrates that using a pre-trained deep language representation model is highly effective for grammatical error detection. Furthermore, MHMLA achieved the

highest F<sub>0.5</sub> on all datasets, outperforming BERT<sub>BASE</sub> by 2.18 points, 1.29 points, 0.81 points, and 1.32 points on FCE, CoNLL14- $\{1,2\}$ , and JFLEG, respectively. The scores for the AvgL model were lower than that for our proposed MHMLA model, meaning that naively using information from layers is not as effective as using MHMLA. These results show that using MHMLA and learning task-specific representations improves the accuracy.

To verify the effect of MHMLA, we examined the F<sub>0.5</sub> value for each head number. We investigated 1, 2, 3, 4, 6, 8, and 12 heads (i.e. the number of heads up to 12 by which the hidden layer size of 768 can be divided). Table 3 shows the F<sub>0.5</sub> values for each number of heads on FCE, CoNLL14- $\{1,2\}$ , and JFLEG datasets. Regarding FCE, the highest F<sub>0.5</sub> score was achieved with 3 heads. For CoNLL14- $\{1,2\}$  and JFLEG, the F<sub>0.5</sub> values were highest with 12 heads, demonstrating that adopting multi-head leads to improved accuracy.

## 6 Analysis of the Effect of MHMLA

The purpose of MHMLA is to construct representations not only from the final layer but also from various layers. Multi-head attention allows the model to jointly attend to information from different representation subspaces at different positions. Therefore, it is considered that increasing the number of heads leads to utilization of information from various layers. Hence, we investigate the effect of the number of heads on each layer by visualizing the averaged score of MHMLA that was calculated by considering the heads  $j$  of Equation



**Fig. 2.** Attention visualization of MHMLA on each dataset using a different number of heads. MHMLA with 8 and 12 heads tends to attend to all layers more or less equally for all datasets.

7 for all layers on test sets of the three datasets: FCE, CoNLL14, and JFLEG.

Figure 2 visualizes the average attention score to each layer of MHMLA for each head. The average attention score is calculated by averaging  $\text{head}_n$  in Equation (8). For all datasets, when there were a fewer numbers of heads, the multi-head attentions learned to attend to different layers but tended to focus on particular layers. For example, as shown in Figure 2(b), multi-head attention with heads of 2, 3, and 4 heads focused more on layers 2 and 3 while hardly attending to layers 5 and 6. Figure 2(b) shows that the same amount of attention is attended to each layer when the number of heads are 8 and 12. In Figure 2(c), attention is sharp, especially with the number of heads being 1, 2, 3, and 4. In contrast, with there are more heads, viz. 8 and 12, attention tended to attend to all layers more or less equally for all datasets. From this visualization, we conclude that our goal of utilizing the information from various layers has been achieved.

## 7 Conclusion

In this study, we investigated the effect of utilizing a deep language representation model ( $\text{BERT}_{\text{BASE}}$ ) pre-trained on large-scale data for grammatical error detection. Simply fine-tuning our  $\text{BERT}_{\text{BASE}}$  model greatly improved  $F_{0.5}$  scores for grammatical error detection task.

Furthermore, we have introduced an approach to learning representations suited for grammatical error detection task from various layers of a pre-trained deep language representation model using MHMLA. Our MHMLA model outperformed previous models for grammatical error detection, establishing new state-of-the-art  $F_{0.5}$  scores. Our analysis demonstrated that we succeeded at learning appropriate representations for a given task using information from different layers.

Future work includes applying MHMLA to other language representation models like Open AI GPT model [1]. Furthermore, with different combination of existing pre-trained language

representation models, we hope to obtain even greater improvements. In addition, we will explore whether our layers learned the same syntactic and semantic roles as a previous work [12], also what exactly self-attention learns at a token-level for grammatical error detection.

## Acknowledgement

This work was partially supported by JSPS Grant-in-Aid for Young Scientists (B) Grant Number JP16K16117.

## References

1. .
2. Al-Rfou, R., Choe, D., Constant, N., Guo, M., & Jones, L. (2019). Character-level language modeling with deeper self-attention. *AAAI*, Association for the Advancement of Artificial Intelligence.
3. Belinkov, Y., Durrani, N., Dalvi, F., Sajjad, H., & Glass, J. (2017). What do neural machine translation models learn about morphology? *ACL*, Association for Computational Linguistics, pp. 861–872.
4. Devlin, J., Chang, M.-W., Lee, K., & Toutanova, K. (2018). BERT: Pre-training of deep bidirectional transformers for language understanding. *arXiv preprint arXiv:1810.04805*.
5. Kaneko, M., Sakaizawa, Y., & Komachi, M. (2017). Grammatical error detection using error- and grammaticality-specific word embeddings. *IJCNLP*, Asian Federation of Natural Language Processing, pp. 40–48.
6. Kasewa, S., Stenetorp, P., & Riedel, S. (2018). Wronging a right: Generating better errors to improve grammatical error detection. *EMNLP*, Association for Computational Linguistics, pp. 4977–4983.
7. Kingma, D. P. & Ba, J. (2015). Adam: A method for stochastic optimization. *ICLR*.
8. Nagata, R. & Nakatani, K. (2010). Evaluating performance of grammatical error detection to maximize learning effect. *COLING*, Coling 2010 Organizing Committee, pp. 894–900.
9. Napoles, C., Sakaguchi, K., & Tetreault, J. (2017). JFLEG: A fluency corpus and benchmark for grammatical error correction. *EACL*, Association for Computational Linguistics, pp. 229–234.
10. Ng, H. T., Wu, S. M., Briscoe, T., Hadiwinoto, C., Susanto, R. H., & Bryant, C. (2014). The CoNLL-2014 shared task on grammatical error correction. *CoNLL: Shared Task*, Association for Computational Linguistics, pp. 1–14.
11. Peters, M., Neumann, M., Iyyer, M., Gardner, M., Clark, C., Lee, K., & Zettlemoyer, L. (2018). Deep contextualized word representations. *NAACL*, Association for Computational Linguistics, pp. 2227–2237.
12. Peters, M., Neumann, M., Zettlemoyer, L., & Yih, W.-t. (2018). Dissecting contextual word embeddings: Architecture and representation. *EMNLP*, Association for Computational Linguistics, pp. 1499–1509.
13. Rei, M. (2017). Semi-supervised multitask learning for sequence labeling. *ACL*, Association for Computational Linguistics, pp. 2121–2130.
14. Rei, M., Felice, M., Yuan, Z., & Briscoe, T. (2017). Artificial error generation with machine translation and syntactic patterns. *BEA*, Association for Computational Linguistics, pp. 287–292.
15. Rei, M. & Søgaard, A. (2019). Jointly learning to label sentences and tokens. *AAAI*, Association for the Advancement of Artificial Intelligence.
16. Rei, M. & Yannakoudakis, H. (2016). Compositional sequence labeling models for error detection in learner writing. *ACL*, Association for Computational Linguistics, pp. 1181–1191.
17. Srivastava, N., Hinton, G. E., Krizhevsky, A., Sutskever, I., & Salakhutdinov, R. (2014). Dropout: A simple way to prevent neural networks from overfitting. *Journal of Machine Learning Research*, Vol. 15, No. 1, pp. 1929–1958.
18. Takase, S., Suzuki, J., & Nagata, M. (2018). Direct output connection for a high-rank language model. *EMNLP*, Association for Computational Linguistics, pp. 4599–4609.
19. Taylor, W. L. (1953). Cloze procedure: A new tool for measuring readability. *Journalism Bulletin*, Vol. 30, No. 4, pp. 415–433.
20. Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A. N., Kaiser, L. u., & Polosukhin, I. (2017). Attention is all you need. In *NIPS*. Curran Associates, Inc., pp. 5998–6008.
21. Wu, Y., Schuster, M., Chen, Z., Le, Q. V., Norouzi, M., Macherey, W., Krikun, M., Cao, Y., Gao, Q., Macherey, K., et al. (2016). Google's neural machine translation system: Bridging the

gap between human and machine translation. *arXiv preprint arXiv:1609.08144*.

22. **Yannakoudakis, H., Briscoe, T., & Medlock, B. (2011).** A new dataset and method for automatically grading ESOL texts. *ACL: Human Language Tech-*

*nologies*, Association for Computational Linguistics, pp. 180–189.

*Article received on 06/03/2019; accepted on 15/03/2019.  
Corresponding author is Masahiro Kaneko.*



# My Word! Machine versus Human Computation Methods for Identifying and Resolving Acronyms

Christopher G. Harris<sup>1</sup>, Padmini Srinivasan<sup>2</sup>

<sup>1</sup> University of Northern Colorado, Dept. of Computer Science,  
USA

<sup>2</sup> University of Iowa, Computer Science Department,  
USA

christopher.harris@unco.edu, padmini-srinivasan@uiowa.edu

**Abstract.** Acronyms are commonly used in human language as alternative forms of concepts to increase recognition, to reduce duplicate references to the same concept, and to stress important concepts. There are no standard rules for acronym creation; therefore, both machine-based acronym identification and acronym resolution are highly prone to error. This might be resolved by a human computation approach, which can take advantage of knowledge external to the document collection. Using three text collections with different properties, we compare a machine-based algorithm with a crowdsourcing approach to identify acronyms. We then perform acronym resolution using these two approaches, plus a game-based approach. The crowd and game-based methods outperform the machine algorithm, even when external information is not used. Also, crowd and game formats offered similar performance with a difference in cost.

**Keywords:** Human computation, crowdsourcing, acronym identification, acronym resolution, gamification.

## 1 Introduction

Acronyms are used in many document collections to abbreviate and stress important concepts. The identification of acronyms and discovery of their associated definitions are essential aspects to tasks such as natural language processing of texts as well as knowledge-based tasks such as information retrieval, named entity resolution, ontology mapping, and question-answering.

Many people think acronyms are standard (e.g. take the first letter of each word and put them together in all capital letters) but there are many

variants from this (e.g., SMART, a German car manufacturer, is an acronym for Swatch + Mercedes + ART; Canola, a type of cooking oil, is an acronym for CANada Oil, Low Acid).

The extraction and resolution of acronyms are not trivial tasks – in many domains, acronyms evolve rapidly. Existing terminological resources and scientific databases cannot keep up to date with the growth of these neologisms. Attempts to manually compose large-scale lexicons of acronym-definition pairs suffer from these same challenges; with such lexicons, information becomes obsolete quickly. Moreover, there are difficulties in the resolution of both ambiguous terms as well as variant forms of the same acronym. For example,

Acronym Finder, the world's largest dictionary of acronyms, includes more than 1 million human-edited definitions and is growing at an average of 5,000 per month [1]; the total effort required to compile this set is estimated to be more than 15,000 hours, a task performed during the last 18 years.

Attending to these shortcomings, this paper evaluates three different non-lexicon approaches to identify and resolve short-form acronyms and their definitions – using machine-based, crowdsourcing-based, and game-based approaches.

We evaluate these methods on three text collections with different characteristics: a collection of news articles, a collection of patents, and a collection of journal articles in chemistry.

Human-based methods possess advantages that machine methods do not; therefore, our examination focuses on examining where human-based methods provide value, and where they do not, as applied to acronym identification and resolution.

We offer the following contributions. First, we take a string-matching algorithm that has demonstrated good results in one domain and test its effectiveness on three new domains. Second, we examine if a higher number of human assessors provides better accuracy with respect to cost. Last, we examine the improvement offered by two human computation approaches – a game-based approach and a crowdsourcing approach – to see if they are better at finding appropriate definitions in the text and if this comes from knowledge outside of the document.

This paper is organized as follows. In the next section, we provide some background and the motivation behind our work. In Section 3, we introduce our research questions and explain our experimental design. In Section 4, we discuss and analyze our results. Following this, we conclude and provide an assessment of our research questions.

## 2 Related Research

The goal here of acronym identification and resolution is to extract pairs of short forms (acronyms) and long forms (their expanded forms or definitions) occurring in text. Much of the work with acronyms is either limited to a specific domain (e.g., biomedical text or government documents) or requires the algorithm to be trained on the corpus before use.

In 2003, the TREC Genomics track [2] began a task that invited acronym identification and definition extraction in biomedical text. This TREC-motivated research encouraged the development of a number of algorithms that performed well against biomedical text it was designed to handle. However, few methods used to examine biomedical text have demonstrated their ability to work effectively on text in other domains, a challenge mentioned in [3]. There have been some attempts to use the broader web to extract definitions of terms, such as that by [4, 5].

Their methods are language independent but are reliant on a large corpus for acronym resolution and may not scale well for documents with rarely-occurring acronyms. In [6], Glass et al. have developed a model that uses Wikipedia to resolve acronyms, but it is dependent on a lexicon of terms. Others, such as [7] have used web mining for detecting acronyms in nursing notes using recurrent neural networking language models (RNNLMs), but this requires a sufficiently large training set in a specialized domain – important because acronyms trained on different domains can lead to dramatically different results.

One challenge in acronym identification and resolution is there are no rules or precise patterns for the creation of acronyms. Moreover, acronyms are ambiguous – the same acronym may refer to two or more different concepts (e.g., *IEM* abbreviates both *immune-electron microscopy* and *interstitial electron model*) and have variant forms (e.g., *NF kappa B*, *NF kB*, *NF-KB*, *NF-kappaB*, and *NFKB* factor for *nuclear factor-kappa B*). Ambiguity and variation present several challenges in text mining approaches since acronyms have not only must be recognized, but their variants must be linked to the same canonical form and be disambiguated, adding to the complexity of acronym recognition through the use of algorithms.

Schwartz and Hearst [8] implemented an algorithm for identifying acronyms that does not need prior training (unsupervised) using parenthetical expressions as a marker of a short form. In their algorithm, an emphasis is on complicated acronym-definition patterns for cases in which only a few letters match. Once a short-form algorithm is found, they use a fixed-sized window on either side of the term to identify the long form candidate. They make a key assumption: either the long-form or short-form acronym appear in parenthesis in the same sentence. Despite the core algorithm being admittedly simple, the authors report 99% precision and 84% recall on the Medstract gold standard. Because of its simplicity and ease of implementation, this algorithm appears appropriate for generalization to collections outside of biomedicine.

Dannélls [9] provided a modified version of the Schwartz and Hearst algorithm, with the advantage of recognizing acronym-definition pairs not indicated by parentheses.



They were able to achieve good precision (above 90%) and recall (above 96%) against four Swedish medical text collections. We use Dannélls algorithm as our machine-based approach; pseudocode of the algorithm is provided in Figure 1.

Human-based approaches can add considerable value to acronym identification and resolution. Humans can adapt to the non-standardized rules commonly found in acronym identification and make use of outside knowledge and apply this to acronym resolution. Despite this, there has no work found in the literature that examines crowdsourcing's ability to detect and resolve acronyms, although some other studies have examined named entity resolution (NER) using the crowd, such as that by Finin et al. on Twitter data [10]. In this paper, we explore the value the crowd provides in acronym identification and resolution.

Game formats provide humans with additional incentives to perform tasks well, such as entertainment, challenge, and recognition (i.e., having a successful player's name added to a leaderboard of top scorers). Despite their potential advantages, we have not found any studies exploring the value of games or crowdsourcing as human computation frameworks for acronym identification and resolution.

Motivated by this context, we study the two tasks of (a) Acronym identification, deciding if strings of text are acronyms and (b) Acronym resolution, mapping the short-form acronym onto its long form. For the first task, we compare a machine-based algorithm with a crowdsourcing-based approach. For the second we compare these two approaches and a game-based one. We present results for three text collections that have different properties.

## 3 Experiment Design

### 3.1 Research Questions

We investigate the following research questions with respect to acronym identification and resolution:

Q1. Can a machine algorithm developed successfully for one domain also show strong results in other domains? (Note that the

algorithm we use does not require training. This is selected intentionally as we wish to stress generalizability to new domains).

- Q2. Are improvements in accuracy achieved with human computation methods over a machine-based algorithm? At what financial cost are improvements, if any, achieved?
- Q3. Does performance improve when more human participants are involved?
- Q4. For acronyms identified correctly by humans, but incorrectly by the algorithm, is the knowledge utilized available in the document, or is it external?
- Q5. How can human computation methods help with the difficult-to-resolve acronyms, e.g., those not easy to resolve algorithmically?

### 3.2 Data

We used 50 documents (for a total of 150), selected randomly from 3 publicly available collections:

- News article documents from TREC collection, disks 4 and 5: LA Times (1989-1990) and Financial Times (1991-1994) newspapers. We used the headline and text/body fields only [11].
- Patent documents come from the MAREC collection, which includes European, Japanese, and US patents, from 2001-2008 [12].
- Chemical Journal articles from four Royal Society of Chemistry journals between 2001 and 2008: Analyst, Journal of Analytical Atomic Spectrometry, Molecular BioSystems, Organic & Biomolecular Chemistry, and Physical Chemistry/Chemical Physics [12].

All documents were in English. Because of their excessive length, if the patent or chemical journal documents exceeded 1000 words, we used the first 1000 words rounded to the end of the paragraph closest to the 1000-word limit. For patents, we used the description field. We did not provide a limit to news articles. Table 1 reports some characteristics of each dataset.

The number of characters and syllables per word from each text collection were not significantly different from each other.

**Table 1.** Characteristics of each collection indicating Mean (M) and Standard Deviation (SD)

Characteristic Measured	Chem Journal M	Chem Journal SD	Patent M	Patent SD	News M	News SD
Number of words	988.33	100.39	1078.00	95.95	1601.00	221.57
Number of sentences	42.67	3.75	96.33	15.48	115.00	20.73
Average number of characters per word	5.75	0.09	5.13	0.12	4.79	0.11
Average number of syllables per word	2.01	0.03	1.84	0.04	1.61	0.04
Average number of words per sentence	22.94	0.43	13.19	1.90	14.81	1.09
Gunning-Fog index	18.91	0.33	16.83	0.94	11.22	0.85
Flesch-Kincaid Grade level	17.06	0.29	14.53	0.86	9.23	0.74
ARI (Automated Readability Index)	17.14	0.29	12.66	0.69	8.55	0.95

**Table 2.** Gold standard data for the acronym identification and resolution tasks

Text Collection	Acronym Identification			Acronym Resolution		
	Assessor 1	Assessor 2	Adjudicated Agreement on Identified Acronyms	Adjudicated Acronyms Requiring Ext. Knowledge to Identify	Adjudicated Agreement on Resolved Definitions	Adj. Acronym Requiring Ext. Knowledge to Resolve
News	189	185	185	11 (5.9%)	183	18 (9.8%)
Patent	269	272	272	28 (10.3%)	266	53 (19.9%)
Chem Journal	335	342	341	26 (7.6%)	320	58 (18.1%)

However, readability analysis shows that the news articles are the easiest to understand for the general public, whereas the chemistry journals are the most difficult.

This was determined consistently both by the Gunning-Fog index (indicating the number of years of formal education a person requires to easily understand the text on the first reading) and the Flesch-Kinkaid and ARI scores.

Both are estimates of the U.S. grade level needed to comprehend the text. Readability is important as human computation methods are being tested.

## 4 Methodology

### 4.1 Gold Standard Data

To create a gold standard acronym list, we had two human assessors who were domain experts each independently evaluate the 150 documents to both identify acronyms and identify their related expansions. Each acronym was counted only once per document and lists for each assessor for each document were adjudicated over any disagreements. The acronym counts (per document) are provided in Table 2.

```

Short-form acronym detection (the identification step):
  Break document into terms and loop through each term;
  IF (the string's length is between 2 and 10 characters) AND (the term contains a string
  of alphabetic, numeric and special characters such as '-' and '/') AND (it contains at
  least two uppercase letters) AND (it does not match terms in a stop list) THEN
    Identify the string as a short-form acronym;
  End;

For long-form acronym detection (the resolution step):
  Examine a term window of size  $\min(|A|+5, |A|^2)$ , where  $|A|$  is the short-form
  acronym length, to the left and to the right of the short-form acronym;
  IF (all the terms within the term window are contained within parentheses) OR (the
  first letter of a term within the term window matches the first letter of the short form
  acronym) THEN
    Identify the string that begins with that term as a potential candidate for a
    long form of that acronym;
  End;

For all long-form acronym candidates:
  IF (the first letter of two consecutive words in the string matches the first letter in two
  or more consecutive capital letters of the short-form acronym) AND (the string does not
  contain a colon, semi-colon, question mark or exclamation mark) AND (the maximum
  length of the string is  $\min(|A|+5, |A|^2)$ , where  $|A|$  is the short-form acronym length)
  AND (the string doesn't contain only upper-case letters) THEN
    Identify the shortest string of terms as our long-form acronym description;
  End;

```

Fig. 1. Pseudocode for the algorithm used for acronym detection and resolution

The figure consists of two side-by-side screenshots of a web interface, both labeled 'Round 3 of 20'.

**Left Screenshot (Identification Task):** The text to be evaluated is: "General service unit [GSU] and regular policemen fired into the air and sealed off the biggest polling station in Lugari [western Kenya] for 30 minutes during by-election voting yesterday. There was no explanation for the siege of Lumakanda divisional headquarters but FORD [Forum for the Restoration of Democracy] Kenya officials claimed it was 'psychological warfare' intended to intimidate voters. There were near-riots at Chekalini polling station when voters hurled stones at KANU [Kenya African National Union] candidate Agili Wawire when he arrived in a convoy of nine vehicles packed with people singing party songs." Below the text, a prompt asks: "Enter the acronym found in the text above, separated by a semi-colon (;). Only one occurrence of an acronym in the above text needs to be entered." There is a text input field and buttons for '<< Back', 'Help', and 'Submit >>'.

**Right Screenshot (Resolution Task):** The text to be evaluated is the same as the left. Below the text, a prompt asks: "Enter the definition of the terms in the text above. Indicate if the definition cannot be determined and if you used personal knowledge outside the document to answer:". There are three rows for definitions: FORD, GSU, and KANU. Each row has a text input field and a radio button for 'Unknown?'. To the right, there are radio buttons for 'Used personal knowledge' with 'Yes' and 'No' options. There are also buttons for '<< Back', 'Help', and 'Submit >>'.

Fig. 2. Screenshots of the acronym identification (left) and resolution (right) tasks provided to crowdworkers

For example, 185 acronyms were found after adjudication for news of which 183 were resolved. We find more acronyms for the journal collection which is consistent with the earlier observation that this collection is the most difficult (of the 3) to read.

## 4.2 Acronym Identification

**Algorithm:** We ran the algorithm on the 150 documents, which output the unique acronym identified and their resolutions. Resolutions not found are marked 'unknown.' For example, common acronym, such as *PM* to represent *afternoon*, are rarely expanded in documents. Therefore, we can track errors in terms of not identifying strings that are acronyms (task 1).

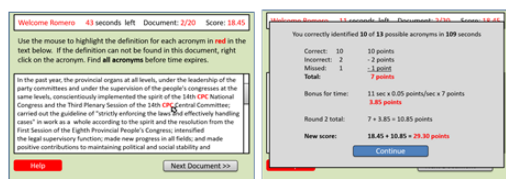
We can also identify errors in resolution (task 2). Figure 1 provides the pseudocode used by the algorithm.

**Crowdsourcing:** We had workers on Amazon Mechanical Turk (MTurk) read each document and identify the acronyms (task 1). Figure 2, left, provides a screenshot.

Each document was processed by nine workers; we tracked the order in which they evaluated the documents.

To determine how the number of crowdworkers affects quality, we calculated consensus judgments as follows.

We took the first three workers submitting results as a set and evaluated majority consensus



**Fig. 3.** Screenshots of the acronym resolution task (left) and the scoring calculations (right) provided to game participants

(2 of 3) and strong majority (3 of 3) consensus decisions.

We repeated this majority and strong majority assessments for sets of the first 5, 7, and 9 workers making submissions. Each worker was paid \$0.03 per document. For each, we evaluated recall, precision, and accuracy calculated against our gold standard.

To reduce noise, we provided “honeypots,” where acronyms were provided in the document and could be found easily.

Workers who did not identify these correctly had their answers removed from the pool, and the task was relisted.

### 4.3 Acronym Resolution

**Algorithm:** Acronym resolution was done in the same process as identification. The analysis was limited to the gold standard acronyms identified (e.g., 183 for the News collection). We found that expanding the window size used by Dannélls did not increase the algorithm’s accuracy. Figure 1 provides the pseudocode used by the algorithm.

**Crowdsourcing:** In a second crowdsourcing run we gave MTurk workers the same documents but with the gold standard acronyms highlighted and asked to resolve them. Figure 2, right, provides a screenshot. Each worker was paid \$0.10 per document. Again, we had nine workers process each document and repeated our assessment of majority consensus and strong majority consensus for the first 3, 5, 7, and 9 workers to make submissions.

Also, we asked each worker to mark whether they used the information solely from the document, or they used common knowledge / outside information. As with the acronym identification task, we provided crowdworkers with

short-form acronyms where the resolution was provided in the document as “honeypots.”

Workers who were unable to identify these had their results removed from the pool and the task was relisted.

**Game:** The game was designed to provide players with a more entertaining and challenging method of resolving acronyms. Figure 3, left, shows a screenshot. Using the same highlighted documents players had to resolve the terms within a specified time limit (2 minutes per document) and were given real-time accuracy scores (Figure 3, right).

A leaderboard was provided for top scorers to enter their names. We listed the game on MTurk and compensated each worker \$0.05 to start the game and encouraged them to continue playing (and resolving acronyms) for as long as possible.

To accommodate acronym resolutions that were similar but not identical to the gold standard, we stemmed the answer provided and the gold standard definition and did a simple character string match on the stemmed terms (e.g., *South African Defence Forces* and *South African Defence Force*, when stemmed, both match the long form of *SADF*).

If they matched, we increased the player’s score. Although this only required for a few participants, later examination showed this technique was 96% accurate at correctly assessing the player’s answer. Accuracy was determined as the proportion of acronyms in our gold standard that were correctly resolved.

## 5 Results and Discussion

### 5.1 Acronym Identification

Table 3 shows the number of acronyms identified by the crowd and machine algorithm for nine assessors.

Table 4 reports accuracy, recall, and precision using majority score for crowdsourcing workers.

Comparing these with scores obtained by the algorithm, the difference for each collection is significant,  $F(2,147) = 4.32$ ,  $p = 0.015$ . We observe that the crowd identified more acronyms than the algorithm in all three collections with greater recall, precision and accuracy scores.

**Table 3.** Acronym identification by the crowd and the machine algorithm

Text Collection	Gold Std (GS)	By Crowd		By Machine Algorithm		
		# Retrieved (5/9 majority)	# Correct (5/9 majority)	# Retrieved	# Correct	# Agreeing w/ Crowd
News	185	198	182	175	144	140
Patent	272	276	259	214	166	162
Chem Journal	341	344	335	295	239	242

**Table 4.** Accuracy (A), precision (P) and recall (R) for the acronym identification task

		News			Patent			Chemistry Journal		
		A	P	R	A	P	R	A	P	R
Crowd	Majority (2/3)	0.959	0.968	0.989	0.952	0.996	0.952	0.994	0.994	0.982
	Strong Majority (3/3)	0.953	0.989	0.984	0.949	1.000	0.952	0.991	0.997	0.982
	Majority (3/5)	0.939	0.989	0.984	0.942	1.000	0.952	0.982	1.000	0.982
	Strong Majority (4/5)	0.939	0.989	0.984	0.942	1.000	0.952	0.980	1.000	0.982
	Majority (4/7)	0.934	0.989	0.984	0.942	1.000	0.952	0.974	1.000	0.982
	Strong Majority (6/7)	0.926	1.000	0.984	0.938	1.000	0.952	0.974	1.000	0.982
	Majority (5/9)	0.919	1.000	0.984	0.938	1.000	0.952	0.974	1.000	0.982
	Strong Majority (7/9)	0.919	1.000	0.984	0.938	1.000	0.952	0.974	1.000	0.982
Machine Algorithm		0.823	0.823	0.778	0.776	0.776	0.610	0.810	0.810	0.701

## 5.2 Acronym Resolution

Table 5 provides accuracy scores for the three methods (algorithm, crowdsourcing, and games). Again, we see that the human computation methods, irrespective of worker set size, do better than the algorithm. Although the crowd format slightly outperformed the game variant; a one-way between-subjects ANOVA found no significant difference in accuracy between them.

We believe the lower performance of the game, although slight, may be due to distractions introduced by the game. There was a significant difference between the crowd/game formats and the machine format. Post-hoc comparisons using the Bonferroni test indicated significant differences between the crowd/game and the machine for the patent and chemical journal collections.

Thus, for these collections, the crowd and game formats were both able to outperform the machine, and this difference was significant for the two

collections even when limiting information used to the document. This indicates that human computation methods are better at pulling definitions from the text even without relying on outside information (Q4). The patterns of gain are similar for games with respect to the machine approach.

The resolution of acronyms benefits from human computation methods most when documents are more challenging to read, implying the resolution of acronyms is more difficult as well (Q2). We also note that the algorithm performs reasonably well in these domains though possibly not as well in the biomedical domains for which it was developed (Q1).

## 5.3 Number of Workers

To answer our research question on the number of workers needed (Q3), we compare the performance scores obtained using all nine

**Table 5.** Table 5: Accuracies for the acronym resolution task

		Document + Ext. Knowledge			From Document Only		
		News	Patent	Journal	News	Patent	Journal
Crowd	Majority (2/3)	<b>0.984</b>	0.941	<b>0.979</b>	<b>0.876</b>	<b>0.732</b>	<b>0.806</b>
	Strong Majority (3/3)	0.973	0.923	0.968	0.870	0.721	0.798
	Majority (3/5)	0.968	0.923	0.971	0.865	0.721	0.801
	Strong Majority (4/5)	0.968	0.912	0.959	0.859	0.713	0.792
	Majority (4/7)	0.957	0.912	0.962	0.859	0.713	0.792
	Strong Majority (6/7)	0.946	0.904	0.956	0.854	0.710	0.789
	Majority (5/9)	0.951	0.904	0.956	0.854	0.710	0.789
	Strong Majority (7/9)	0.941	0.904	0.956	0.849	0.710	0.789
Game	Majority (2/3)	0.951	<b>0.949</b>	0.977	0.849	0.724	0.798
	Strong Majority (3/3)	0.930	0.934	0.959	0.832	0.717	0.786
	Majority (3/5)	0.930	0.938	0.959	0.832	0.721	0.786
	Strong Majority (4/5)	0.908	0.919	0.950	0.816	0.710	0.783
	Majority (4/7)	0.930	0.919	0.979	0.816	0.710	0.783
	Strong Majority (6/7)	0.908	0.915	0.974	0.811	0.706	0.780
	Majority (5/9)	0.908	0.915	0.974	0.811	0.706	0.780
	Strong Majority (7/9)	0.908	0.915	0.971	0.811	0.706	0.780
Machine Algorithm		0.778	0.610	0.701	0.778	0.610	0.701
Increase in score by best human computation method over machine algorithm:		0.205	0.338	0.279	0.097	0.122	0.105
ANOVA F-value		16.81	63.85	217.85	5.28	23.19	22.45
Significance		0.056	0.015	0.005	0.159	0.041	0.043

assessors with scores using only the first three assessors. In both tasks, we found no significant difference in the quality of results.

This demonstrates that we need only a few assessors from the crowd to obtain quality results. This is similar to the findings by Snow et al. on a separate study of non-experts in NLP tasks [14]. Cutting the number of assessors from 9 to 3 can reduce assessment costs by two thirds.

This, coupled with no measurable gain in quality reinforces our reluctance to use more than three members of the crowd (or game participants) in similar identification and resolution tasks.

Using a paired t-test, we also found no significant difference in quality between the crowd and game approaches. For the acronym resolution

task, we spent \$135.00 for the nine crowd assessors and half that amount to obtain similar quality in the game with a similar number of assessors. This does not consider the fixed cost required for game development – for short term evaluation; this cost is a factor; for an evaluation over a much longer period, it becomes a trivial cost.

Some acronyms, particularly those in patents, could not be ascertained in the text by any approach and required background knowledge of the domain. The largest reason for acronyms being missed by humans in the identification task is that people confused abbreviations (e.g., Sr. for senior) with acronyms. Machine-based techniques rarely make this type of mistake.

Overall, humans were able to resolve 93%, 86% and 96% of the acronym identification errors made by the machine approach and 44%, 22% and 33% of the acronym resolution errors (on acronyms not requiring external knowledge) made by the machine approach for the news, patent, and chemical journal collections, respectively (Q5). We believe this may be due to the machine algorithm evaluated rules on acronym identification without flexibility, whereas humans were flexible and resolved acronyms that did not follow common rules (such as *ATM machine*, where *machine* confounds the algorithm's ability to resolve the long-form equivalent).

Humans provided value when there were several candidate definitions, and the correct one required context to resolve, such as resolving *PM* when both *Post Meridian* and *Prime Minister* appear in the document. Humans also provided value in acronym resolution when definitions in the text contained embedded short-form acronyms (e.g., *PRESTO* is defined as *Precursory Research for Embryonic S&T Program*, and *S&T* is defined elsewhere in the same document as *Science and Technology*).

## 6 Conclusion and Perspectives

We have applied two separate approaches (a machine algorithm and a crowdsourcing approach) to three different publicly-available text collections in an algorithm identification task.

We applied these two approaches, plus a game-based approach, to a task to find the long-form acronym definition to the short-form acronym in the same text collections. The machine algorithm we used, designed for biomedical text, was unable to obtain the same accuracy, precision and recall rates for any of our three text collections.

We evaluated that an increase in accuracy of 10-30% can occur when non-expert human computation methods are used. For those acronyms incorrectly resolved by the algorithm, we found that many of these errors did not rely on external information from the human participant (e.g., using *ETA* for *Estimated Time of Arrival*); although external knowledge plus the ability for humans to resolve acronyms with the information in the text provided the best results.

For acronym identification and resolution tasks, we found adding additional assessors did not provide an improvement in accuracy, precision or recall. We also found that most algorithmic errors were a result of the inability to evaluate exceptions to specified rules, which humans can do relatively well. In the horizon, more advanced techniques for identifying and resolving acronyms may be able to mimic human decision-making, closing the gap between human computation and machine approaches.

It should be noted that the costs were initially much higher to set up the gamification interface over a standard interface on a crowdsourcing platform like MTurk. However, because we paid a flat fee to game participants, making the game "sticky," or capable of captivating a user's attention for a sustained period of time, gamification can save money in the long run, since people keep participating as long as their interest is maintained.

Adding stickiness to the game interface had the adverse effect of making it more complex. This may have served as a distraction from the task of resolving acronyms (as observed from the game's slightly lower accuracy scores when compared to the crowd interface), mitigating some of the gamification interface's potential. We, therefore, believe that the crowd interface provides the best overall approach across each collection for acronym identification and resolution.

In future work, we plan to look at low-resource languages, particularly those that do not use capital letters. We also plan to incorporate phonotactics; short forms with no vowel or "y" are more likely to be acronyms (since they are less likely to be real words); the same goes for consonant combinations that are not normally found in the English language.

One particularly tricky area considered for a follow-up study is acronyms composed of initial syllables of words. These tend to be phonotactically well-formed words and often become so lexicalized that people no longer realize they are acronyms. *Scuba* and *laser* in English, *nazi* in German and *kolkhoz* in Russian are of this type.

## Appendix A

### Perl program used for machine algorithm

```
#!/usr/bin/perl
use File::Find;
use Lingua::EN::Sentence qw(get_sentences);
use Cwd('abs_path');
use Data::Dumper;
use strict;

my $stop_dir = abs_path($ARGV[0]);
my % stop_list_term = ('a' => 1, 'on' => 1, 'of' => 1, 'in' => 1, 'the' => 1, 'at' => 1, 'an'
=> 1, 'for' => 1.);
my $stop_list_regex = join('|', keys % stop_list_term);
$stop_list_regex = qr/\b$stop_list_regex\b/;
my %acronyms;

sub get_acronym {
    my ($s) = @_;
    my $i = -1;
    my @acrs = ();
    foreach my $w((split(/\s+/, $s))) {
        $i++;
        my $nxt;
        next
        if ($w =~ /[:;!?]/);
        foreach my $stop (keys % stop_list_term){
            if ($w =~ /^$stop$/i){
                $nxt=1;
                last;
            }
        }
        next if ($nxt);
        my $weight = $# {[(($w =~ /[:upper:]/g))]} +1;
        $w =~ s/([(){}?*"'\w+)(["'(){}?]*[., ])?/?$2/g;
        if ($weight== 2 && length $w >= 2 && length $w <= 10 && $w =~ /^[a-z0-9-
V]+$/i){
            $w =~ s/^(.+)\w$/1/;
            my $acr = {};
            $acr->{word} = $w;
            $acr->{index} = $i;
            $acr->{weight} = $weight;
            @{$acr->{letters}} = ();
            foreach ( split(/([[:upper:]]/), $w)) {
                if ($_) { push @{$acr->{letters}}, lc $_; }
            }
            push @acrs, $acr;
        }
    }
    return \@acrs;
}

sub is_start_word_as_acr {
    my ($word, $acr) = @_;
    my $fl = substr($word, 0, 1);
    if (lc $fl eq lc substr($acr, 0, 1)){
        return 1;
    }
    return 0;
}

sub check_for_if {
    my ($s, $acr, $docno) = @_;
    my @result;
    my $term_window = ((length $acr->{word})+5 < ((length $acr->{word})^2)?
((length $acr->{word})+5):(length $acr->{word})^2;
    my @ words = split(/\s+/, $s);
    if (join(' ', @words{$acr->{index}+1 .. $#words}) =~ /\b(?:\w+)?(?:\w+)?\b/){
        push @{$acronyms{$docno}->{$acr->{word}}}, $1;
    }
    my @words = map {s/([(){}?*"'\w+)(["'(){}?]*[., ])?/?$2/g; $_} @words;
    my $left_boundary = ($acr->{index}-$term_window >= 0)?($acr->{index}-
$term_window):0;
    my $right_boundary = ($acr->{index}+$term_window <= $#words)?($acr-
->{index}+$term_window):$#words;
    my @ ls_if = @words{$left_boundary .. $acr->{index}-1};
    my @ rs_if = @words{$acr->{index}+1 .. $right_boundary};
    foreach (@ls_if){
        if(is_start_word_as_acr($_, $acr->{word})){
            push @result, [@$ls_if];
            last;
        }
    }
    foreach(@rs_if) {
        if (is_start_word_as_acr($_, $acr->{word})) {
            push @result, [@$rs_if];
            last;
        }
    }
    return \@ result;
}

sub some_checks_for_description {
    my ($candidate, $acr, my $fl_acr) = @_;
    if ($candidate =~ /(?:!|,|/ or $candidate !~ /(?:lower:|/
or $candidate =~ $acr->{word}or $candidate =~ /\s+/) {
        return 0;
    }
    if (length $acr->{word}-$fl_acr >= 2){
        return 0;
    }
    ## print "xx ", (grep {!/$stop_list_regex$/} split(/ /, $candidate)), "\n";
    if (scalar (grep {!/$stop_list_regex$/} split(/ /, $candidate)) <2){
        return 0;
    }
    my %tmp = ();
    foreach (split(/ /, $candidate)){
        if(exists $tmp {$_}) {
            return 0;
        }
        $tmp {$_}=1;
    }
    return 1;
}

sub get_description {
    my($acr, $fl) = @_;
    my @ candidates = ();
    foreach my $longform (@$fl) {
        my @fl_acr;
        my %w_for_skip = ();
        my($acr_l, $_exit, $candidate, $last_term) = ({}, 0, "", "");
        my @ tmp_candidates = ();
        while (!$_exit) {
            my ($skip, $start, $check) = (0,0,0);
            $candidate = "";
            @fl_acr = map {($_ = /\s+/)?(lc $_=>1):({$_=>0})} split(/ /, $acr-
->{word});
            my $acr_l = shift @fl_acr;
            while ((values %$acr_l)[0] == 0) {
                $acr_l = shift @fl_acr;
            }
            foreach my $fl_term (@$longform) {
                ## print $acr->{word}, " $fl_term\n";
                next if(exists $w_for_skip {$fl_term});
                if ((exists $acr_l->{(lc substr($fl_term, 0, 1))}) && $start && $skip == 0
&& $fl_acr >= 0 && !exists $stop_list_term {$fl_term}) {
                    ## print "candidate = $candidate\n";
                    $w_for_skip {$last_term}=1;
                    if (some_checks_for_description($candidate, $acr, $fl_acr)){
                        push @tmp_candidates, $candidate;
                        ## print Dumper \@candidates;
                    }
                    $check = 1;
                    last;
                }
            }
            $last_term = $fl_term;
        }
        if (exists $acr_l->{(lc substr($fl_term, 0, 1))}) &&
$acr_l->{(lc substr($fl_term, 0, 1))} == 1) {
            if ($skip) {$skip=0;}
            unless($start) {$start=1;}
            ## print "add $fl_term\n";
            $candidate = "$fl_term ";
            ## print $candidate, "\n";
            $acr_l = shift @fl_acr;
            next;
        }
        if ((exists $acr_l->{(lc substr($fl_term, 0, 1))}) && $start &&
```



```

    $acr_l->{(lc substr($if_term, 0, 1)) == 0} || $skip == 1){
    unless($skip) {$skip=1;}
    # print "add1 $if_term \n";
    $candidate .= "$if_term ";
    }
    if (exists $stop_list_term($if_term) && $start && $#fl_acr >= 0){
    # print "add2 $if_term \n";
    $candidate .= "$if_term ";
    }
    }
    if ($check == 0) {
    $exit = 1;
    }
    }
    if (some_checks_for_description($candidate, $acr, $#fl_acr) {
    push @candidates, $candidate;
    # # printDumper \@candidates;
    }elseif($#tmp_candidates >= 0) {
    push @candidates, pop @tmp_candidates;
    }
    }
    return \@candidates;
}

sub acronym {
    my ($docno, $lines) = @_;
    $lines = ~s/\n/\./mg;
    my $sentences = get_sentences($lines);
    foreach my $s (@$sentences){
        if ($s =~ /-/ ) {
            $s = (split(/-/ , $s))[1];
        }
        next if $s =~ /([[:lower:]]g)/ == -1;
        my $acrs = get_acronym($s);

        foreach my $acr (@$acrs) {
            my $if_for_check = check_for_if($s, $acr, $docno);
            my $descr;
            if ($acr->{word} && $#{if_for_check}>1) {
                $descr = get_description($acr, $if_for_check);
            }else{
                push @{$acronyms{$docno}->{$acr->{word}}}, 'undefined';
                next;
            }
            if ($#{ $descr } >= 0) {
                foreach (@$descr) {
                    push @{$acronyms{$docno}->{$acr->{word}}}, $_;
                }
            }else{
                push @{$acronyms{$docno}->{$acr->{word}}}, 'undefined';
            }
        }
    }
}

sub wanted {
    my $fn = $File::Find::name;
    my($doc, $docno, $lines);
    if (open(F, $fn)) {
        while (my $str = <F>){
            if ($str =~ m#<DOCNO>\s?(\\s+)\s?</DOCNO>#){
                $docno = $1;
            }
            if ($str =~ /<TEXT>/) { $doc = 1; }
            if ($str =~ /</TEXT></DOC>/){
                $doc = 0;
            }
            if ($docno) {
                if ($lines =~ /\w+/) {
                    acronym($docno, $lines);
                }
                $docno = 0;
            }
            $lines="";
        }
        if ($doc) {
            if ($str =~ /<[^>]+>.+<[^>]+>/){
                next;
            }
            $lines .= $str;
        }
    }
    close F;
}

```

```

    }
    find(&wanted, $top_dir);

    foreach my $docno (keys %acronyms){
        foreach my $acr(keys %{$acronyms{$docno}}){
            my $defined = 0;
            foreach my $descr (@{$acronyms{$docno}->{$acr}}){
                if ($descr ne 'undefined') { $defined = 1; }
            }
            my %tmp = ();
            unless($defined) {
                print "docno $acr - undefined\n";
                next;
            }
            foreach my $descr (@{$acronyms{$docno}->{$acr}}){
                $descr =~ s/\s//g;
                if ($defined && $descr ne 'undefined') {
                    unless(exists $tmp{$descr}) {
                        print "docno $acr - $descr\n";
                        $tmp{$descr}=1;
                    }
                }
            }
        }
    }
}

```

## References

1. Molloy, M. (2010). *Passes the 1 million definition milestone*.
2. Hersh, W.R. & Bhupatiraju, R.T. (2003). TREC genomics track overview. *TREC*, pp. 1–10.
3. Moon, S., McInnes, B., & Melton, G.B. (2015). Challenges and practical approaches with word sense disambiguation of acronyms and abbreviations in the clinical domain. *Healthcare informatics research*, Vol. 21, No. 1, pp. 35–42. DOI: 10.4258/hir.2015.21.1.35.
4. Sánchez, D. & Isern, D. (2011). Automatic extraction of acronym definitions from the Web. *Applied Intelligence*, Vol. 34, No. 2, pp. 311–327. DOI: 10.1007/s10489-009-0197-4.
5. Menaha, R., Barkavi, M., Prashanthini, P.G., & Narmadha, R. (2016). A web based approach: Acronym Definition Extraction. *International Research Journal of Engineering and Technology (IRJET)*, Vol. 3, No. 2, pp. 1199–1206.
6. Glass, M.R., Chowdhury, M.F.M., & Gliozzo, A.M. (2017). Language Independent Acquisition of Abbreviations. arXiv preprint arXiv:1709.08074.
7. Kirchhoff, K. & Turner, A.M. (2016). Unsupervised Resolution of Acronyms and Abbreviations in Nursing Notes Using Document-Level Context Models. *Proceedings of the Seventh International Workshop on Health Text Mining and Information Analysis*, pp. 52–60.
8. Schwartz, A.S. & Hearst, M.A. (2002). A simple algorithm for identifying abbreviation definitions in

- biomedical text. *Biocomputing*, pp. 451–462. DOI: 10.1142/9789812776303\_0042.
9. **Dannélls, D. (2006).** Automatic acronym recognition. *Proceedings of the Eleventh Conference of the European Chapter of the Association for Computational Linguistics: Posters & Demonstrations*, pp. 167–170.
  10. **Finin, T., Murnane, W., Karandikar, A., Keller, N., Martineau, J., & Dredze, M. (2010).** Annotating named entities in Twitter data with crowdsourcing. *Proceedings CSLDAMT '10 Proceedings of the NAACL HLT 2010 Workshop on Creating Speech and Language Data with Amazon's Mechanical Turk*, pp. 80–88.
  11. **Voorhees, E. & Harman, D. (2000).** Overview of the sixth text retrieval conference (TREC-6). *Information Processing & Management*, Vol. 36, No. 1, pp. 3–35. DOI: 10.1016/S0306-4573(99)00043-6.
  12. **Lupu, M., Piroi, F., Huang, X., Zhu, J., & Tait, J. (2009).** Overview of the TREC 2009 chemical IR track. York University Downsview (Ontario).
  13. **Snow, R., O'Connor, B., Jurafsky, D., & Ng, A.Y. (2008).** Cheap and fast-but is it good?: evaluating non-expert annotations for natural language tasks. *Proceedings of the conference on empirical methods in natural language processing*, pp. 254–263.

Article received on 18/01/2019; accepted on 04/03/2019.  
Corresponding author is Christopher G. Harris.

# Ontology-based Extractive Text Summarization: The Contribution of Instances

Murillo Lagranha Flores, Elder Rizzon Santos, Ricardo Azambuja Silveira

Federal University of Santa Catarina,  
Department of Informatics and Statistics, Florianópolis,  
Brazil

murillo.flores@posgrad.ufsc.br, {elder.santos,ricardo.silveira}@ufsc.br

**Abstract.** In this paper, we present a text summarization approach focusing on multi-document, extractive and query-focused summarization that relies on an ontology-based semantic similarity measure, that specifically explores ontology instances. We employ the DBpedia Ontology and a theoretical definition of similarity to determine query-sentence and sentence-sentence similarity. Furthermore, we define an instance-linking strategy that builds the most accurate sentence representation possible while achieving a better coverage of sentences that can be represented by ontology instances. Using primarily this instances linking strategy, the semantic similarity measure and the Maximal Marginal Relevance Algorithm (MMR), we propose a summarization model that is capable of avoiding redundancy from a more fine-grained representation of sentences, due to their representation as ontology instances. We demonstrate that our summarizer is capable of achieving compelling results when compared with relevant DUC systems and recently published related studies using ROUGE metrics. Moreover, our experiments lead us to a better understanding of how ontology instances can be used to represent sentences and what is the role of said instances in this process.

**Keywords.** Extractive text summarization, ontologies, ontological instances.

## 1 Introduction

Text Summarization is the task of creating a shorter version of a document or a set of documents while keeping most of the informational content present in these documents. Automatic Text Summarizers are usually classified with regards to how they construct the final summary

as either extractive or abstractive [12]. In extractive summarization, the summary is built by concatenating textual units (usually paragraphs or sentences) extracted from the original documents. Due to its conceptual simplicity and the guarantee that the sentences used in summary will be at least as legible as the sentences in the original documents extractive summarization has been a very prominent approach in automatic text summarization for the last decades.

Constructing an extractive summary that covers most of the information present in the original documents while achieving a significant reduction in length is essential to avoid redundancy. Ontology-based summarizers proposed so far explored the use of concepts as a proxy to represent the semantics of sentences, successfully avoiding redundancy and therefore achieving great results in generating extractive summaries.

However, due to their inability to distinguish between different references to the same concept, which reduces their ability to evaluate the semantics of sentences, ontology-based extractive summarizers that only explore concepts have the tendency to leave relevant sentences out of the summary for considering them redundant, when in fact they hold references to different instances of the same concept. The use of manually built ontologies makes the problem more severe, due to their reduced number of concepts.

In this paper we propose to use ontology instances to represent the semantics of sentences, attacking the problem mentioned above.

**Input sentences**

- S1 For Manchester United, the reigning English champs, this is nothing new.
- S2 Manchester City, on the other hand, hasn't won the English title since 1968.

**Representation using Concepts - R1**

- 1 [dbo:SoccerClub, dbo:Country]
- 2 [dbo:SoccerClub, dbo:Country]

**Representation using instances - R2**

- 1 [dbr:Manchester\_United\_F.C., dbr:England]
- 2 [dbr:Manchester\_City\_F.C., dbr:England]

---

dbo: dbpedia.org/ontology  
dbr: dbpedia.org/resource

**Fig. 1.** Representation of sentences using concepts and instances defined in a ontology

Figure 1 depicts the advantages of using instances to represent the semantics of a sentence. Sentences S1 and S2 are referencing two distinct football teams from the same city, and positioning them with regards to their past performance on the English football championship. In conjunction, they are comparing and making an argument about both teams performances.

When these sentences are represented as a vector of concepts, as can be seen in R1, their representations are identical. When they are represented as a vector of instances instead, as seen in R2, their representation changes and comes closer to the real semantic differences between them.

With the goal of improving the quality of summaries built by extractive query-focused text summarizers in mind, we present an ontology-instances based summarization model. Our model uses an automatic annotation tool to link sentences to instances defined in an ontology, then uses these instances to represent the sentences and finally a semantic similarity measure to calculate

the similarity between two sets of instances. We experiment on the DUC2005 dataset.

## 2 Representing Sentences as Concepts

To evaluate the impact that representing sentences as concepts have on the process of detecting redundant sentences, we calculated the overlap between the concepts representing distinct sentences of a summary from two summary sets. We used the DUC2004 task 2 dataset on this evaluation. In this dataset model summaries are manually created summaries, and peer summaries have been limited to include only summaries created by participating systems.

We started by employing a system for automatic annotation of DBpedia instances on text, DBpedia spotlight [5], to identify references to instances on each summary. Because the instances described in the DBpedia ontology are linked to other ontologies, we then grouped the concepts that appeared in the *rdf:type* of each instance by ontology. After that, we selected the first concept that was listed as the *rdf:type* of each group as the concept that best represented that mention on the text. With that, we created vectors of concepts that appeared in each sentence of each summary, per ontology. Those lists were considered the representation of each sentence as a vector of concepts. We consider the final result of this process to be similar to what an ontology-based summarizer that only employ concepts to represent sentences would achieve.

We calculated the intersection between the vectors of concepts representing the sentences of each summary. We classified the results in **total intersection** when the same vector of concepts represented at least two sentences of the same summary, and **partial intersection** when at least one concept appeared in two distinct vectors representing sentences of the same summary. Table 1 shows the final results. We only included results for the DBpedia and the Schema.org ontologies, as those are the larger ones linked to DBpedia instances and therefore can generate a more diverse representation of sentences.

We found that peer summaries tend to have a lower total and partial intersections than model summaries as can be seen in table 1.

**Table 1.** Percentage of documents with two or more sentence representations matching totally or partially on DUC 2004

Documents set	Total intersection	Partial intersection
<b>Peer</b>		
dbpedia	0.346	0.747
schema.org	0.343	0.705
<b>Model</b>		
dbpedia	0.601	0.870
schema.org	0.626	0.845

These results demonstrate that it is common to reference the same concept more than once in model summaries, created by humans. Therefore, using only concepts to detect and avoid redundancy has the potential to remove sentences that appear to be important in human-created summaries. It appears that a more granular semantic representation, that can compare the differences between sentences more precisely can achieve better results. Corroborates to this idea the fact that peer summaries have lower total and partial intersections.

### 3 Base Model

We use the MMR algorithm as our base model [3]. At each iteration, the algorithm selects a sentence to extract and include in the summary, until the desired length is reached. The sentence selected is always the one that is (i) more similar to the query and (ii) less redundant when compared with the previously selected sentences. We choose the MMR algorithm as our base model because it is specifically tailored for extractive query-focused summarization and can easily be extended to incorporate the advantages of a better similarity measure capable of comparing sentences and query. MMR is defined as in expression 1:

$$MMR \stackrel{def}{=} \max_{D_i \in R/S} \left[ \alpha(sim_1(D_i, Q)) - (1 - \alpha) \max_{D_j \in S} sim_2(D_i, D_j) \right], \quad (1)$$

where  $Q$  is a query,  $R$  is a documents collection (cluster),  $S$  is the subset of documents in  $R$  already selected,  $R/S$  is the set of yet unselected documents and  $sim_1$  and  $sim_2$  are similarity metrics.

## 4 Semantic Similarity Using Instances

We extend MMR through the definition of a semantic similarity measure capable of calculating query-sentence and sentence-sentence similarity that can be used by that algorithm.

### 4.1 Representing Sentences as Instances

In order to determine a sentence's relevance to a specific query using ontology instances a representation of each one of them using said instances must be constructed. To this end, we employ an Instances Linking System (ILS). Instances linking systems will take snippets of text, in our case a sentence, as input and output a list of ontology instances that are mentioned in the input.

One typical problem faced by instances linking systems is the absence of detected mentions due to either a reduced number of instances defined in the underlying ontology or some inefficiency of the ILS. Instances linking systems might allow the configuration of a confidence parameter, that determines the minimum level of confidence that the ILS must have in order to link a mention, to address the problem of ILS inefficiency. Ensuring that all sentences have a valid representation is fundamental to guarantee that an instances based summarizer will be able to operate correctly, therefore we devise an approach to deal with the problem mentioned above based on the possibility of configuring a confidence parameter, as shown in algorithm 1, where  $ILSLink$  is a function that links instances at a given level of confidence using the ILS.

**Algorithm 1** IL with variable confidence

---

```

1: Input  $T$ : Text from a sentence or query.
2: Input  $c$ : Initial confidence value.
3: procedure LINK( $T, c$ )
4:    $s \leftarrow ILSLink(T, c)$ 
5:   if  $s = \emptyset$  &  $c \geq 0.1$  then
6:      $c \leftarrow c - 0.1$ 
7:      $s \leftarrow Link(T, c)$ 
8:   return  $s$ 

```

---

**4.2 Similarity Between sets of Instances**

We defined sentence-sentence and query-sentence semantic similarity using their representations as ontology instances. Inspired by the work of [11] the semantic similarity between two sets of instances is defined as the average of the maximal similarity between the instances representing each one of the sets, as shown in expression 2:

$$Sim(S_1, S_2) = \frac{1}{2} \left[ \frac{\sum_{i_1 \in S_1} \max_{i_2 \in S_2} Sim(i_1, i_2)}{|S_1|} + \frac{\sum_{i_2 \in S_2} \max_{i_1 \in S_1} Sim(i_1, i_2)}{|S_2|} \right], \quad (2)$$

where  $Sim$  is a semantic similarity measure between two ontology instances. We define and describe the measure used in this work in section 4.3. This definition assumes a symmetrical contribution of each one of the instance sets under comparison.

When used in conjunction with the algorithm defined in section 4.1 this definition ensures that the contribution to the overall similarity added by instances linked at a given level of confidence will not decrease with lower confidence values. In other words, its maximal similarity will not decrease with the addition of instances that are less strongly related to the sentence in question. It is worth noting, however, that the addition of said instances does increase the number of instances representing each sentence which might increase the denominator in each side of expression's 2 sum.

**4.3 Similarity between Instances**

We use the theoretical definition of similarity presented by [7] to define the semantic similarity measure used in this work. According to [7] "The similarity between A and B is measured by the ratio between the amount of information needed to state the commonality of A and B and the information needed to fully describe what A and B are" and can be expressed by the following equation:

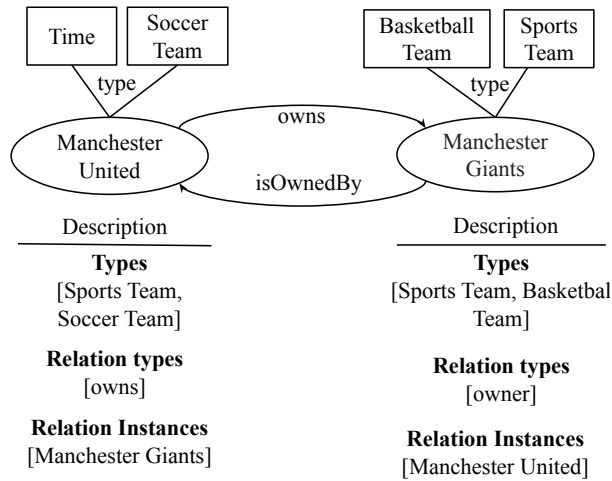
$$sim(A, B) = \frac{\log P(common(A, B))}{\log P(description(A, B))}. \quad (3)$$

We believe that the relations that an ontology instance holds with other instances contain valuable semantic information about it. We also believe that an instance's types - the concept that they are an instance of - hold a significant amount of semantic information about it. We define that the description of an instance is formed by its types and relations with other instances, for the sake of computing its semantic similarity with other instances. Furthermore, we define that each relation will add two pieces of information to the description, separately: its type and the instance it connects to. Therefore, the description of an instance will be formed by three distinct categories of information: its **types**, its **relation types** and its **relation instances**. Figure 2 depicts two instances of different concepts that are related to each other and their description as it would be used to compute their semantic similarity.

To apply Lin's [7] definition of similarity to this work, we first define that the probability of any given component of the description of an instance is given by the probability of that component being present at a randomly selected instance of the ontology, as shown in equation 4. The probability of a relation type, for instance, is the probability of a relation of that type being present on a randomly selected instance of the ontology:

$$P(component) = \frac{count(\text{instances with component})}{count(\text{total number of instances})}. \quad (4)$$

We expand the definition presented in equation 4 to define the similarity of a description category



**Fig. 2.** Instances description example

(types, relation types and relation instances) as in equation 5, where  $cat$  is a function that returns all components of one of the categories of information in the description of the instance passed as a parameter:

$$\frac{2 * \left( -1 * \sum_{c \in (cat(A) \cap cat(B))} \log P(c) \right)}{\left( -1 * \sum_{c \in cat(A)} \log P(c) \right) + \left( -1 * \sum_{c \in cat(B)} \log P(c) \right)}. \quad (5)$$

Finally, the overall similarity is defined as the average of category similarities, as expressed in equation 6, where  $Sim_{types}$  is the **types** similarity,  $Sim_{relTypes}$  is the **relation types** similarity and  $Sim_{relInst}$  is the **relation instances** similarity:

$$sim(A, B) = \frac{1}{3} \left[ Sim_{types} + Sim_{relTypes} + Sim_{relInst} \right]. \quad (6)$$

As an example, table 2 presents the similarity between instances of the 2014 DBpedia Ontology calculated following the previous definitions.

The results in table 2 align well with the values expected from a similarity measure that follows the assumptions defined in [7]. Maximum similarity is achieved when an instance is compared against

**Table 2.** Similarity between 2014 DBpedia Ontology Instances

#	Measure	Value
1	$sim(\text{L.A. Lakers}, \text{L.A. Lakers})$	1
2	$sim(\text{L.A. Lakers}, \text{G.S. Warriors})$	0.6248
3	$sim(\text{L.A. Lakers}, \text{N.E. Patriots})$	0.3958
4	$sim(\text{N.E. Patriots}, \text{S. Seahawks})$	0.6301
5	$sim(\text{L.A. Lakers}, \text{Spider Man})$	0.0363

itself (line 1). Teams of the same league - Los Angeles Lakers and Golden State Warriors are in the same league, as well as New England Patriots and Seattle Seahawks - have a higher similarity when compared against each other than when compared against a team in the other league (lines 2, 4 and 3). The similarities between teams in the same league have close values for both leagues (lines 2 and 4). Moreover, the similarity between a sports team and a fictional character is an order of magnitude smaller than between two sports teams on different leagues (lines 3 and 5).

## 5 Our Model

We extend the MMR algorithm by employing the instances linking strategy and the similarity measures defined in the previous section, as shown in figure 3.

First, instances are linked to the input documents and the query. When linking instances to the query either a fixed minimum confidence or the strategy discussed in section 4.1 are used. After that, the input documents and the query are segmented into sentences.

The MMR algorithm then uses these sentences and the instances linked to them to extract sentences and build the summary following the definition in expression 7, where  $S_Q$  are the query sentences,  $S_D$  are the documents sentences,  $S_S$  are the subset of the document sentences already selected,  $S_D/S_S$  are the yet unselected document sentences and  $sim$  is the similarity measure defined in section 4.2.

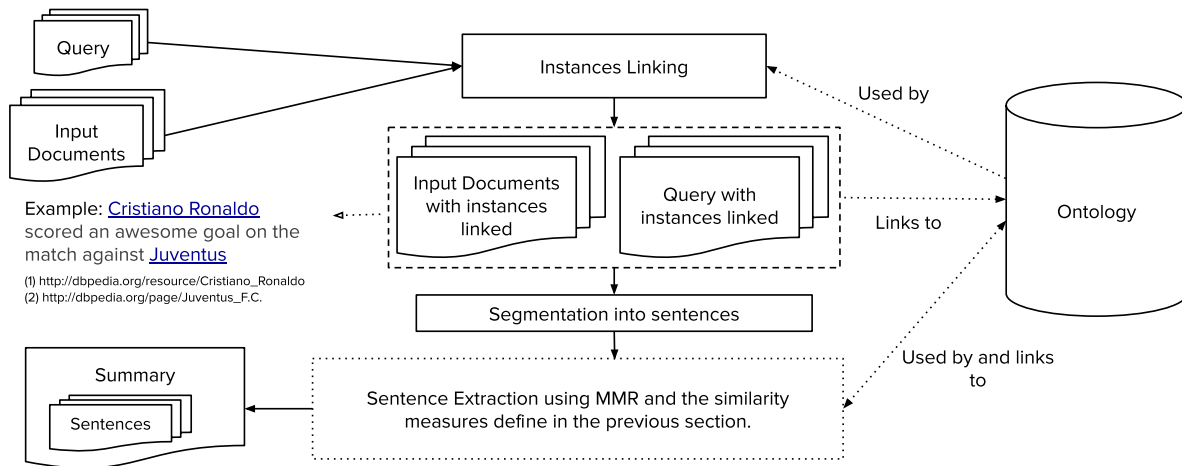


Fig. 3. The full architecture of our model

$$MMR \stackrel{def}{=} \max_{S_i \in S_D / S_S} \left[ \alpha(sim(S_i, S_Q)) - (1 - \alpha) \max_{S_j \in S} sim(S_i, S_j) \right]. \quad (7)$$

## 6 Related Works

Different authors have used ontologies in numerous approaches to address the extractive summarization problem.

[13] used an ontology to create a graph where each ontological concept of the document becomes a vertex and every relation between concepts becomes an edge. The most “central” sentences on that graph are extracted and establish the summary. [1] used the YAGO ontology to evaluate sentences considering a feature that expresses the sentence’s popularity and pertinence, called *entityRank*. Later, sentences are extracted using a variation of MMR strategy [3]. [4] described the adopted techniques and the design of a system called Texminer, which uses ontologies in a very similar approach to the one followed by [14]. [15], heavily influenced by [8] applies an ontology to represent sentences as sets of concepts and to compute the similarity between the sentences. Closely related to our work [11] derived an approach for extractive multi-document

query-focused summarization based on a semantic similarity measure that employed the WordNet Taxonomy as its knowledge-based. The authors enhanced the similarity measure with named entity semantic relatedness inferred from Wikipedia.

Different approaches that did not employ ontologies also addressed the multi-document extractive summarization problem. [2] proposed a query-focused approach based on a weighted archetypal analysis (wAA), a multivariate data representation using matrix factorization and clustering. [9] also proposed a query-focused approach suggesting to focus on three different considerations: 1) relevance, 2) coverage and 3) novelty in a probabilistic modeling framework.

Previous studies on extractive summarization have only used ontologies to capture the hierarchy of concepts in a specific domain, effectively using them as a taxonomy. Ontology instances have not been explored so far, and we are the first ones to use them to represent sentences as a way to compare these sentences semantically and enhance the summarizer’s performance.

## 7 Experiments

In this section, we report the experiments conducted to evaluate the effectiveness of our



proposed model in multi-document query-focused extractive summarization.

## 7.1 Experimental Settings

We used the DUC 2005 dataset for evaluation. The DUC 2005 dataset is formed by 50 document clusters, each containing between 25 and 50 different documents on a specific topic. Each cluster has on average 31 documents and 20,236 words. The desired number of words in the summaries is 250. For each document set, between four and ten model summaries are available. This dataset was specifically created for the evaluation of multi-document query-focused summarizers.

To evaluate the summaries generated by our system quantitatively and compare them against baselines summaries as well as against summaries generated by closely related systems we use the Rouge family of metrics [6]. Rouge metrics are the *de-facto* standard in extractive summary evaluation, being widely used in the existing literature. The assessment of the quality of a summary carried out by Rouge-n metrics is based on existing model summaries (usually generated by humans) and the co-occurrence of n-grams between those model summaries and the summaries under evaluation. The evaluation follows the definition in expression 8:

$$ROUGE - N = \frac{\sum_{S \in \{Ref.Summ.\}} \sum_{gram_n \in S} Count_{match}(gram_n)}{\sum_{S \in \{Ref.Summ.\}} \sum_{gram_n \in S} Count(gram_n)}, \quad (8)$$

where  $N$  is the length of the N-gram,  $Count(gram_n)$  is the number of n-grams in the reference summary and  $Count_{match}(gram_n)$  is the maximum number of n-grams co-occurring in the summary being evaluated and a set of reference summaries (Ref. Summ.).

## 7.2 Implementation

To conduct the experiments we implemented our proposed model selecting the DBpedia Ontology as our base ontology. The 2014 DBpedia Ontology was built by knowledge extracted from Wikipedia and has more than four million instances defined in it [5].

We used DBpedia Spotlight to link DBpedia ontology instances to text. DBpedia Spotlight is capable of linking instances through different surface forms and with a configurable disambiguation confidence [10].

We experimented with two different variations of our system, one using a fixed value for instances linking disambiguation confidence on both documents and query and one using a variable value for the query, as described in section 4.1. We ran each variation with three initial confidence values - 0.3, 0.6, 0.9 - and three MMR  $\alpha$  values - 0.3, 0.5, 0.7 - totalizing 18 different experimental runs.

As for computing Rouge metrics, we used the same ROUGE-1.5.5.pl Perl script used to compute the scores in the original DUC2005 competition. The parameters used were also the same ones used by DUC2005<sup>1</sup>.

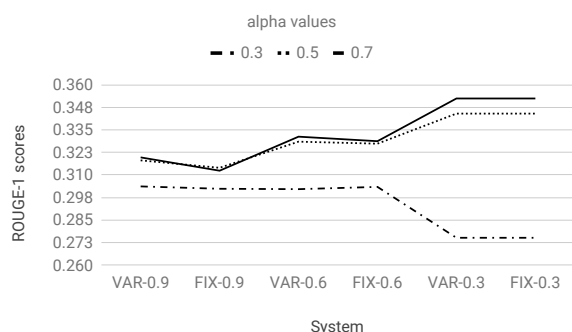
## 7.3 Results

We evaluated the quality of the summaries generated by our systems using Rouge-1 and Rouge-2 as these perform better in multi-document summarization evaluation [6].

The systems name notation used in the figures describing results is defined as follows: Each system name is formed by a prefix and a suffix, separated by a dash ("-"). The prefix indicates whether that version of the system used a fixed (FIX) or a variable (VAR) confidence value when linking instances to the query. The suffix indicates the initial confidence value of the system. As an example, VAR-0.6 indicates that that version of the system ran with a variable confidence value (to link instances to the query, as described in section 4.1) starting from 0.6.

<sup>1</sup>ROUGE-1.5.5.pl -n 2 -x -m -2 4 -u -c 95 -r 1000 -f A -p 0.5 -t 0 -d

Figure 4 shows the ROUGE-1 scores obtained by all systems, with three different values of the MMR parameter  $\alpha$  configured. This parameter controls the balance between query relevance and summary diversity when selecting sentences. The figure shows that all systems presented better results as the instances annotation confidence decreased for values of  $\alpha$  greater than or equal to 0.5 when query relevance had a more significant impact on sentence selection. With the  $\alpha$  value set to 0.3 the opposite occurred - the results decreased as the confidence decreased, with a particularly acute drop between systems with confidence configured to 0.6 and 0.3. It is also worthwhile to note that except for FIX-0.9 all systems achieved their best results with  $\alpha$  set to 0.7. These results indicate that if more relevance is given to the query, the more instances are annotated in the documents, the better. If summary diversity is given more importance when selecting sentences, more instances annotated in the documents might lead to worst results.

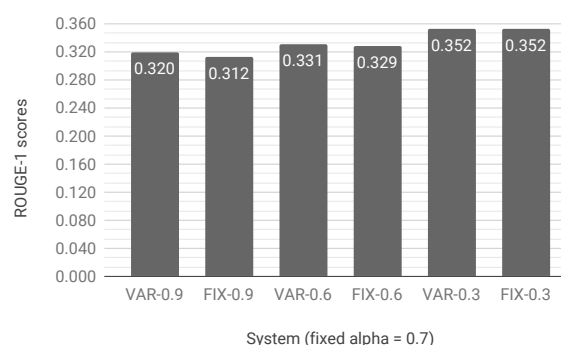


**Fig. 4.** Rouge-1 scores per system, with three different values of alpha

One possible explanation for achieving better performance with lower confidence and higher alpha values lives on the length difference between the query and the documents. Because the query is very short when compared to the documents, the extra instances it gets with lower values of confidence compensates the noise introduced by the extra, possibly irrelevant, instances linked to the documents. This extra instances will increase sentence extraction performance

significantly when alpha is configured to a value that gives query relevance more importance than summary diversity - or at least equals to.

Figure 5 shows that the systems with variable decreasing confidence in instances annotation on the query achieved better results than the versions with fixed confidence at the same starting level of confidence, in two occasions for a fixed  $\alpha$  of 0.7. That corroborates with the explanation that more instances annotated on the query are more relevant to increase performance with higher values of  $\alpha$ .

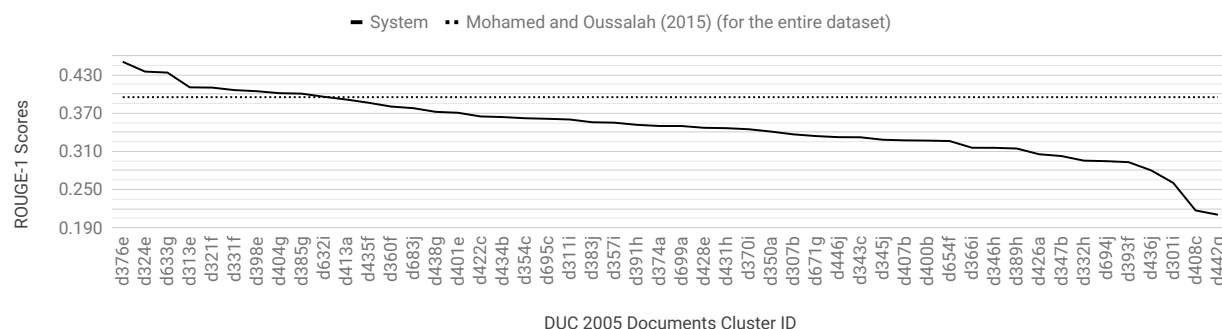


**Fig. 5.** Rouge-1 scores per system, with a fixed value of alpha (0.7)

Table 3 presents a comparison between the average of DUC2005 systems, closely related works and the results obtained by the best variant of our systems, VAR-0.3 system with  $\alpha$  set to 0.7. All systems were experimented in the DUC2005 dataset. Our system outperforms the average DUC2005 systems in both compared metrics, but it also falls behind all the other systems under comparison in both metrics.

**Table 3.** Comparison between the average of DUC2005 systems, closely related works and our results

System	Rouge-1	Rouge-2
Avg. DUC2005 Systems	0.3434	0.0602
Luo et al. [9]	0.3728	0.0807
Canhasi et al. [2]	0.3945	0.0797
Mohamed et al. [11]	0.3949	0.0693
This work	0.3524	0.0639



**Fig. 6.** Rouge-1 scores per DUC2005 document cluster, obtained by the best variant of our system.

We also analyzed the Rouge-1 Scores obtained by the best variant of our system in all 50 DUC2005 document clusters. The results are shown in figure 6, ordered by decreasing Rouge-1 scores from left to right. To help visualize the quality of the results we also plot a line representing the best result shown in table 3 [11] for the entire dataset. As can be seen in the figure, at first the results achieved are above that line. They then decrease in a way that resembles a linear descent with a sudden drop at the end.

These results indicate that it is possible to achieve great results using instances to represent sentences and the techniques described in section 5, but further analysis is required to understand what's preventing the system from performing better on the clusters where performance is falling above the compared best.

We can draw from the conducted experiments that ontology instances can contribute to boosting the performance of extractive multi-document query-focused summarizers, by enhancing sentence-query similarity comparison and therefore helping identify sentences that are more relevant to the query. The fact that all versions of our summarizer presented better (or at least equal) results when an effort to enhance the query representation was made, by varying the instances linking confidence parameter as described in section 4.1, is an empirical evidenced of that. Following the same idea, we can also understand that the performance of summarizers based on ontology instances is highly dependent on the

quantity and semantic coverage of the instances defined in the ontology and on the quality of the instances linking process. Better algorithms and similarity metrics can remediate an excess of irrelevant instances linked to the query and the documents, but they cannot remediate a lack of instances.

## 8 Conclusion

We proposed to use ontology instances to build an extractive query-focused multi-document summarization model, as a way to achieve a more fine-grained representation of the semantics of sentences, and avoid the problem of over-pruning sentences due to a limited semantic representation. We showed that when ontology concepts are used to represent the semantics of sentences, human-created summaries have more sentences with overlapping representations than automatically generated ones.

We extended the MMR algorithm to build our model, through an instance linking strategy with variable linking confidence and a similarity measure based on ontology instances. We experimented on the DUC2005 dataset and concluded that although representing sentences as ontology instances can help boost summarization performance further analysis is still needed to achieve better results.

## References

1. Baralis, E., Cagliero, L., Jabeen, S., Fiori, A., & Shah, S. (2013). Multi-document summarization based on the Yago ontology. *Expert Syst. Appl.*, Vol. 40, No. 17, pp. 6976–6984.
2. Canhasi, E. & Kononenko, I. (2014). Weighted archetypal analysis of the multi-element graph for query-focused multi-document summarization. *Expert Syst. Appl.*, Vol. 41, No. 2, pp. 535–543.
3. Carbonell, J. & Goldstein, J. (1998). The use of MMR, diversity-based reranking for reordering documents and producing summaries. *Proceedings of the 21st Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, SIGIR '98, ACM, New York, NY, USA, pp. 335–336.
4. Hipola, P., Senso, J. A., Leiva-Mederos, A., & Dominguez-Velasco, S. (2014). Ontology-based text summarization. The case of Texminer. *Library Hi Tech*.
5. Lehmann, J., Isele, R., Jakob, M., Jentzsch, A., Kontokostas, D., Mendes, P., Hellmann, S., Morsey, M., van Kleef, P., Auer, S., & Bizer, C. (2014). DBpedia - a large-scale, multilingual knowledge base extracted from Wikipedia. *Semantic Web Journal*.
6. Lin, C.-Y. (2004). ROUGE: A package for automatic evaluation of summaries. *Proc. ACL workshop on Text Summarization Branches Out*, pp. 10.
7. Lin, D. (1998). An information-theoretic definition of similarity. *Proceedings of the Fifteenth International Conference on Machine Learning*, ICML '98, Morgan Kaufmann Publishers Inc., San Francisco, CA, USA, pp. 296–304.
8. Lin, H. & Bilmes, J. (2010). Multi-document summarization via budgeted maximization of sub-modular functions. *Human Language Technologies: The 2010 Annual Conference of the North American Chapter of the Association for Computational Linguistics*, HLT '10, Association for Computational Linguistics, Stroudsburg, PA, USA, pp. 912–920.
9. Luo, W., Zhuang, F., He, Q., & Shi, Z. (2013). Exploiting relevance, coverage, and novelty for query-focused multi-document summarization. *Know.-Based Syst.*, Vol. 46, pp. 33–42.
10. Mendes, P. N., Jakob, M., García-Silva, A., & Bizer, C. (2011). DBpedia spotlight: Shedding light on the web of documents. *Proceedings of the 7th International Conference on Semantic Systems*, I-Semantics '11, ACM, New York, NY, USA, pp. 1–8.
11. Mohamed, M. A. & Oussalah, M. (2015). Similarity-based query-focused multi-document summarization using crowdsourced and manually-built lexical-semantic resources. *2015 IEEE Trustcom/BigDataSE/ISPA*, volume 2, pp. 80–87.
12. Nenkova, A. & McKeown, K. (2012). *A Survey of Text Summarization Techniques*. Springer US, Boston, MA, pp. 43–76.
13. Ramezani, M. & Feizi-Derakhshi, M.-R. (2015). Ontology-based automatic text summarization using FarsNet. *Advances in Computer Science: an International Journal*, Vol. 4, No. 2, pp. 88–96.
14. Umbrath, W., Wetzker, R., & Hennig, L. (2008). An ontology-based approach to text summarization. *Web Intelligence and Intelligent Agent Technology, IEEE/WIC/ACM International Conference on*, Vol. 03, No. undefined, pp. 291–294.
15. Wu, K., Li, L., Li, J., & Li, T. (2013). Ontology-enriched multi-document summarization in disaster management using submodular function. *Information Sciences*, Vol. 224, pp. 118 – 129.

Article received on 30/01/2019; accepted on 04/03/2019.  
Corresponding author is Murillo Flores.

# Ontology-driven Text Feature Modeling for Disease Prediction using Unstructured Radiological Notes

Gokul S. Krishnan, Sowmya Kamath S.

National Institute of Technology Karnataka, HALE Lab,  
Department of Information Technology, Surathkal,  
India

gsk1692@gmail.com, sowmyakamath@nitk.edu.in

**Abstract.** Clinical Decision Support Systems (CDSSs) support medical personnel by offering aid in decision-making and timely interventions in patient care. Typically such systems are built on structured Electronic Health Records (EHRs), which, unfortunately have a very low adoption rate in developing countries at present. In such situations, clinical notes recorded by medical personnel, though unstructured, can be a significant source for rich patient related information. However, conversion of unstructured clinical notes to a structured EHR form is a manual and time consuming task, underscoring a critical need for more efficient, automated methods. In this paper, a generic disease prediction CDSS built on unstructured radiology text reports is proposed. We incorporate word embeddings and clinical ontologies to model the textual features of the patient data for training a feed-forward neural network for ICD9 disease group prediction. The proposed model built on unstructured text outperformed the state-of-the-art model built on structured data by 9% in terms of AUROC and 23% in terms of AUPRC, thus eliminating the dependency on the availability of structured clinical data.

**Keywords.** Healthcare informatics, unstructured text, disease prediction, ontologies, natural language processing.

## 1 Introduction

Modern healthcare applications are largely aided by Clinical Decision Support Systems (CDSSs) which mostly involve predictive and preventive analytics applications for betterment of patient healthcare delivery. Digital revolution has led to continuous generation and streaming of huge amount of data in the form medical records,

radiology reports, prescriptions, clinical notes, scan images, etc. The continuing research in developing CDSS by making use of the huge amount of generated medical data indicate the significant efforts to augment the decision making made by medical caregivers.

Several Machine Learning (ML) and Data Mining based CDSSs have been developed over the years, helping caregivers make informed decisions and timely interventions during critical conditions of patients. Some major CDSS applications like mortality prediction [5, 4, 8, 16], hospital readmission prediction [13], length of stay prediction [19, 1], disease-specific prediction [9, 10, 20], generic disease or ICD9<sup>1</sup> diseases/group prediction [17, 15, 6] and disease coding of discharge summaries [2, 3, 7, 21] have gathered significant research interest in the field of healthcare and biomedicine.

Most of the existing CDSSs developed over the years largely depend on the availability of structured patient data in the form of Electronic Medical Records (EMRs) or Electronic Health Records (EHRs), which is suited for usage in developed countries due to the large scale adoption of EHRs in hospitals. However, hospitals in developing countries still depend on clinical notes of free and unstructured text format. Therefore, there is a crucial need for alternative methods to develop effective CDSSs without relying on structured hospital or patient data.

<sup>1</sup>ICD-9-CM: International Classification of Diseases, Ninth Revision, Clinical Modification

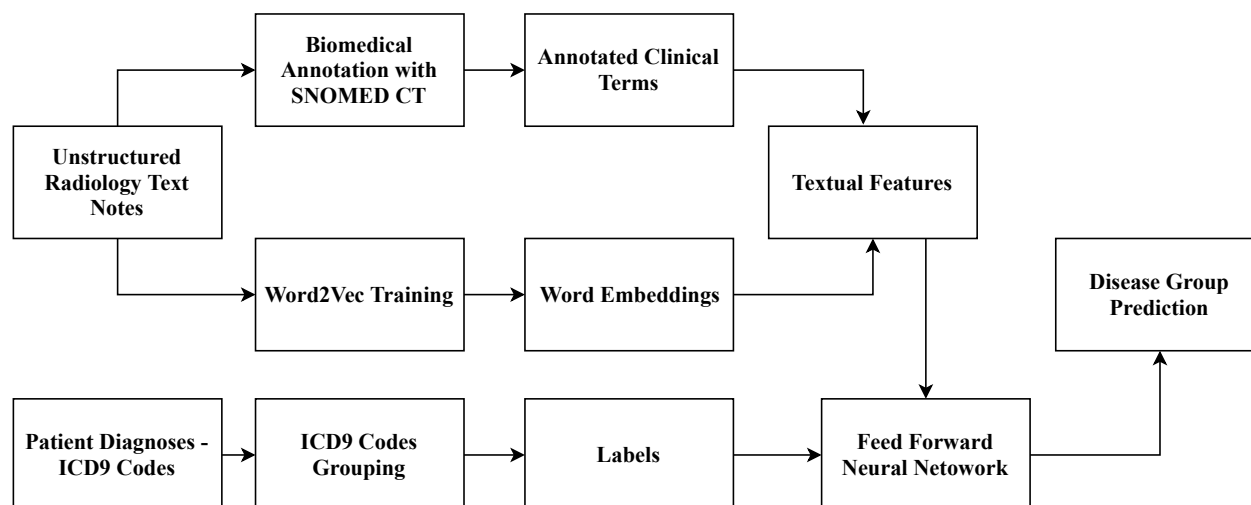


Fig. 1. Proposed approach

Purushotham et al. [17] benchmarked three prediction models - ICU mortality, ICD9 group prediction and hospital readmission on large healthcare data using Super Learner and Deep Learning techniques. The ICD9 group prediction task presented in their work is a generic disease prediction task that can be a good CDSS for caregivers and can also be considered as a prior step for ICD9 code prediction.

This work was also based on structured patient data and the benchmarking was performed on various feature subsets and the best performing model used 105 features which included a variety of feature values such as input events (fluids and medications given through IV), output events (urinary and rectal output), prescribed medicines and other ICU events like chartevents (readings noted during ICU stay) and labevents (lab test results).

As deep learning models can learn to classify or regress from numerous raw feature values quite well, their approach achieved good results. However, in a real world scenario, to measure all these values, convert them into a structured EHR form and then to predict an outcome, a significant time delay is inevitable, which might lead to worsening of patient condition. Thus, disease prediction models that can make predictions with high accuracy with low latency, which can

predict with high accuracy over lower test data are the need of the hour.

In this paper, a feature modeling approach that adopts the concepts of word embedding and ontologies to model features effectively to train and build a neural network based model to predict diseases (ICD9 disease group) is presented. We show the results of the benchmarking study of our proposed model (modeled on unstructured radiology notes) against the current state-of-the-art model (built on structured clinical data), where our model performed on par with the state-of-the-art model.

The rest of this paper is structured as follows: Section 2 describes the proposed approach in detail, followed by experimental results and discussion in 3 and 4, after which we conclude the paper with prospective future work and directions.

## 2 Materials and Methods

The overall workflow of the proposed approach for disease group prediction is as depicted in Figure 1. Radiology reports in unstructured text format from the open and standard MIMIC-III [11] dataset were used for this study. MIMIC-III contains data about 63,000 ICU admissions of around 46,000 patients admitted in Beth Israel Hospital, New York, USA between 2001 and 2012.

From the 'NOTEEVENTS' table, only the Radiology notes were extracted for this study. Overall, 1,94,744 radiology text reports generated during 45,512 admissions of 36,447 patients were included for the study. Often, a patient may be diagnosed with multiple diseases in the same admission, hence, it is necessary for the prediction to be a multi-label prediction task. Therefore, for each radiology report, all disease groups were considered as labels and given binary values - 0 (if the disease was not present) and 1 (if the disease was present).

**Table 1.** Dataset and Cohort Characteristics

Feature	Total Records
Patients	36,447
Admissions	45,512
Radiology Reports	194,744
Sentences	539,466
Words	45,755,992
Average word Length of Report	235
Unique Diseases	2,593
Disease Groups	21

## 2.1 Preprocessing & Textual Feature Modeling

The radiology reports text corpus were first subjected to a basic Natural Language Processing (NLP) pipeline consisting of tokenization and stopping processes. The tokenization process breaks down the clinical text corpus into tokens and the stopping process filters out unimportant words (stop words) from the corpus. The preprocessed tokens corpus is then fed into a SNOMED-CT ontology based annotator to annotate and extract clinical and biological/biomedical terms. SNOMED-CT ontology [18] is an ontology that provides a vocabulary of clinical/biomedical terms and helps extract associated concepts from the preprocessed radiology report corpus. We used the Open BioMedical Annotator [12] for this purpose, after which 4,366 unique clinical/biomedical terms were obtained.

The presence or absence of each extracted clinical/biomedical term, represented as binary values, is considered as a textual feature representation.

The preprocessed corpus is also used to train a Word2Vec [14] word embedding model to extract the word embedding features from the corpus. The skipgram model of Word2Vec was used for training the corpus as this model takes word ordering into consideration and is effective with infrequent words as well. The skipgram Word2Vec model is trained with a dimension size of 500 and initial learning rate of 0.01. The word embeddings were extracted such that each report is represented as 1 x 500 vector. The word embedding features were further concatenated with the extracted clinical/biomedical term features with binary values for each report indicating its presence (1) or absence (0) in the respective reports and the feature matrix was then standardized to values between -1 and 1. These features are used for training the neural network model for disease prediction.

## 2.2 ICD9 Disease Code Grouping

The ICD9 disease codes of patients' diagnoses were retrieved from the 'DIAGNOSES\_ICD' table of MIMIC-III dataset and the labels were grouped as per available standards<sup>2</sup> and as previously followed by state-of-the-art work [17]. A total of 2,593 unique ICD9 disease codes were accordingly grouped into 21 ICD9 disease groups (as shown in Table 2). As a patient can suffer from multiple diseases, we consider the ICD9 group prediction task as a binary classification of multiple labels. Therefore, 21 different labels (disease groups) were considered with possible binary values: 0 (for absence of the disease) and 1 (for presence of the disease). The radiology reports of the selected cohort did not have any case of external injury (E-codes), hence, for the 194744 x 4966 feature matrix, 20 ICD9 disease groups were considered as labels to train the neural network model, which is described in Section 2.3.

<sup>2</sup>Available online [http://tdrdata.com/ipd/ipd\\\_SearchForICD9CodesAndDescriptions.aspx](http://tdrdata.com/ipd/ipd\_SearchForICD9CodesAndDescriptions.aspx)

**Table 2.** ICD9 Disease Grouping - Dataset Statistics

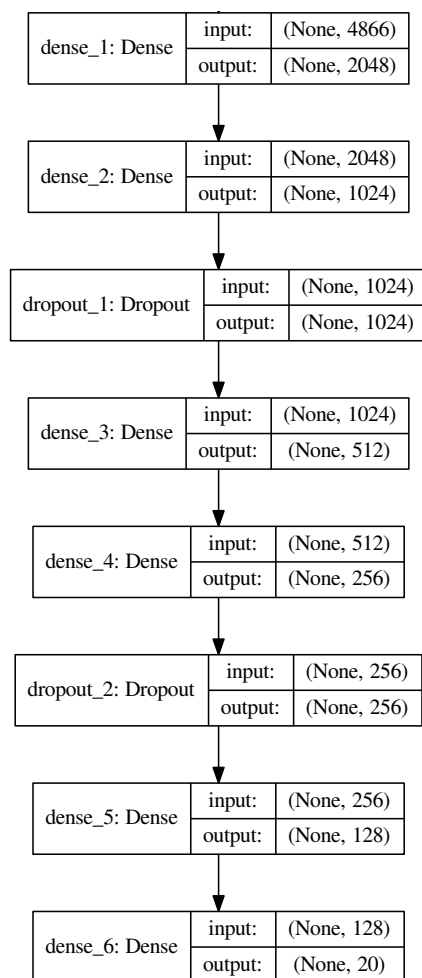
ICD9 Group (Label)	ICD9 Code Range	Description	Occurrences (MIMIC-III)	Occurrences (Study sample)
1	001 - 139	Infectious and Parasitic Diseases	13,686	7,289
2	140 - 239	Neoplasms	9,457	68,734
3	240 - 279	Endocrine, Nutritional, Metabolic, Immunity	44,671	34,668
4	280 - 289	Blood and Blood-Forming Organs	16,345	128,357
5	290 - 319	Mental Disorders	15,053	80,337
6	320 - 389	Nervous System and Sense Organs	13,327	62,963
7	390 - 459	Circulatory System	96,749	65,149
8	460 - 519	Respiratory System	31,984	152,159
9	520 - 579	Digestive System	25,206	107,656
10	580 - 629	Genitourinary System	21,884	84,346
11	630 - 677	Pregnancy, Childbirth, and the Puerperium	524	89,305
12	680 - 709	Skin and Subcutaneous Tissue	5,791	601
13	710 - 739	Musculoskeletal System and Connective Tissue	7,951	29,046
14	740 - 759	Congenital Anomalies	3,429	39,703
15	760 - 779	Conditions Originating in the Perinatal Period	19,605	10,115
16	780 - 789	Symptoms	13,965	71,784
17	790 - 796	Nonspecific Abnormal Findings	3,034	20,803
18	797 - 799	Ill-defined and Unknown Causes of Morbidity and Mortality	947	6,664
19	800 - 999	Injury and Poisoning	30,573	108,867
20	V Codes	Supplementary Factors	50,318	100,310
21	E Codes	External Causes of Injury	14,003	0

### 2.3 Disease Prediction Model

The feature matrix with both word embedding features and ontologically extracted term-presence features, along with ICD9 group labels are next used for training a neural network based prediction model. A Feed Forward Neural Network (FFNN) architecture was used to build the prediction model, which is depicted in Figure 2.

The input layer consists of 2048 neurons with input dimension as 4966 (number of input features); 4 hidden layers with 1024, 512, 256 and 128 neurons respectively and finally an output layer with 20 neurons, each representing an ICD-9 disease group. To prevent overfitting, two dropout layers, with a dropout rate of 20% was also added to the FFNN model (see Figure 2).





**Fig. 2.** Feed Forward Neural Network Model for ICD9 Group Prediction

As this is a binary classification for multiple labels, the loss function used for the FFNN was binary cross entropy. Stochastic Gradient Descent (SGD) was used as the optimizer and a learning rate of 0.01 was used. The *tanh* activation function was used as the input and hidden layer activation functions as the feature matrix values are standardized to the range -1 and 1. The major hyperparameters for the FFNN model – the optimizer, learning rate of the optimizer and the activation function, were tuned empirically over several experiments using the GridSearchCV

function in Python sklearn library. Finally, the output layer activation function was a sigmoid function, again as the classification was binary for each of the 20 labels. Training was performed for 50 epochs and then the model was applied to the validation set to predict disease groups after which the results were observed and analyzed.

### 3 Experimental Results

To evaluate the performance of the proposed model, standard metrics to measure machine learning models were considered – accuracy, precision, recall, F-score, Area Under Receiver Operating Characteristic curve (AUROC), Area Under Precision Recall Curve (AUPRC) and Matthew's Correlation Coefficient (MCC). We performed the evaluation of these metrics on a sample-wise basis, i.e., the predicted and actual ICD9 disease groups were compared and analyzed for each radiology report. It can be observed from the Table 3 that, the proposed model achieved promising results: AUPRC of 0.74 and AUROC of 0.84. The accuracy of 0.77 and precision of 0.80 also indicate effective prediction performance of the proposed approach.

We also compared the performance of the proposed approach against an existing ICD9 disease group prediction model, a Multimodal Deep Learning (MMDL) architecture put forward by Purushotham et al. [17], which is built on structured patient data. As the number of records and features under consideration for both the studies are different, it is to be noted that this is a metric based comparison. During validation experiments, it was observed that the proposed approach significantly outperformed against the state-of-the-art method by 23% considering the AUPRC metric and 9% in terms of AUROC. To encourage other comparative studies, certain additional experiments were made.

We also provide the Recall & F-Score performance as well as the MCC values of the proposed model over our easily reproducible patient cohort dataset. The model showed good results in these experiments, achieving a recall of 0.77, F-score of 0.77 and MCC value of 0.50. It is to be noted that our method

**Table 3.** Experimental Results

Parameter	Proposed Approach	Purushotham et al. [17]
Total admissions	45,512	38,425
Type of Data	Unstructured Text	Structured data
AUROC	$0.84 \pm 0.01$	$0.77 \pm 0.01$
AUPRC	$0.74 \pm 0.01$	$0.60 \pm 0.02$
Accuracy	0.77	*
Precision	0.80	*
Recall	0.77	*
F-Score	0.77	*
MCC	0.50	*

\* Metric not reported in the study

performed better than the state-of-the-art [17], despite being built on a significantly larger number of patient admission data than the state-of-the-art approach (see Table 3). Further, we achieve this performance using only textual features and we did not make use of structured patient data or processed information from any kind of structured data to model the radiology reports of patients. Thus, there is an added advantage that the conversion from unstructured text data to a structured representation can be ignored, thereby achieving huge savings in person hours, cost and other resources.

## 4 Observations and Discussion

From our experiments, we observed a very high requirement and potential of developing prediction based CDSSs using unstructured text reports rather than the usage of structured patient data and EHRs. The proposed text feature modeling was effectively able to capture the rich and latent clinical information available in unstructured radiology reports, and the neural network model used these features to effectively learn disease characteristics for prediction. The Word2Vec model generated word embedding features and the extracted terms using the Open Biomedical Annotator and SNOMED-CT ontology further enhanced the semantics of the textual features

thereby enabling the FFNN to generalize better and learn the feature representation well resulting in effective prediction performance of the proposed approach.

The high values of metrics AUPRC of 0.74 and AUROC of 0.84 in comparison to the state-of-the-art model's (built on structured data) AUPRC of 0.60 and AUROC of 0.77 respectively, is an indication that the unstructured text clinical notes (radiology reports in this case) contain abundant patient-specific information that can be used for predictive analytics applications and that the conversion process from unstructured patient text reports to structured data can be eliminated thereby saving huge person hours, cost and other resources. Moreover, the proposed approach also eliminates any dependency on structured EHRs, thus making it suitable for deployment in developing countries.

Few other insights into the challenges also came to light during our experiments. We found that the initial data preparation approach used for this study could be improved, as it resulted in some conflicting cases during training. In the MIMIC-III dataset, the radiology reports do not have a direct link to ICD9 disease code and to overcome this, we designed a data preparation approach for extracting ICD9 codes from the DIAGNOSES.ICD table and then assign them to all patients with the

same SUBJECT.ID and HADM.ID in the radiology notes corpus.

A negative effect of this approach was that, in some cases, the ICD9 disease codes/groups were assigned to radiology text reports were not related to that particular disease. This could have affected the model's accuracy, due to assignment of conflicting labels to textual features of radiology notes. Nevertheless, the model achieved promising results, as is evident in the values of metrics like precision (0.80), accuracy (0.77), F-score (0.77), recall (0.77) and MCC (0.50). The AUROC (0.84) and AUPRC (0.74) values also show that a disease prediction model built on unstructured radiology text reports can perform well as a real-world CDSS application for hospitals and caregivers.

## 5 Conclusion and Future Work

A neural network based model for predicting ICD9 disease groups from unstructured radiology text reports was presented in this paper. The approach is built on a feature set modeled using Word2Vec to generate word embedding features and also SNOMED-CT ontology based annotator to extract clinical/biomedical terms and concepts whose presence or absence are considered as features. The ICD9 disease codes were categorized into 21 standard groups and then used to train a binary classifier for multi-label prediction.

A FFNN architecture was used to train the classifier and the prediction model was validated and benchmarked against state-of-the-art ICD9 disease group prediction model. The experiments highlighted the promising results achieved by the proposed model, outperforming the state-of-the-art model (built on structured patient data) by 23% in terms of AUPRC and 9% in terms of AUROC. This indicates that the proposed approach can be considered as a viable alternative, as it eliminates the dependency on structured clinical data, thereby ensuring that hospitals in developing countries with low EHR adoption rate can also utilize effective CDSS in their functioning.

As part of future work, we plan to address the issues observed in the designed data preparation strategy, and enhance it by sorting

out the disease group assignment problems. We also intend to explore other feature engineering techniques to further optimize topic and feature modeling representations for their effect on disease prediction.

## Acknowledgements

We gratefully acknowledge the use of the facilities at the Department of Information Technology, NITK Surathkal, funded by Govt. of India's DST-SERB Early Career Research Grant (ECR/2017/001056) to the second author.

## References

1. Appelros, P. (2007). Prediction of length of stay for stroke patients. *Acta Neurologica Scandinavica*, Vol. 116, No. 1, pp. 15–19.
2. Ayyar, S., Don, O., & Iv, W. (2016). Tagging patient notes with ICD-9 codes. *Proceedings of the 29th Conference on Neural Information Processing Systems*.
3. Berndorfer, S. & Henriksson, A. (2017). Automated diagnosis coding with combined text representations. *Studies in health technology and informatics*, Vol. 235, pp. 201.
4. Calvert, J., Mao, Q., Hoffman, J. L., Jay, M., Desautels, T., Mohamdlou, H., Chettipally, U., & Das, R. (2016). Using electronic health record collected clinical variables to predict medical intensive care unit mortality. *Annals of Medicine and Surgery*, Vol. 11, pp. 52–57.
5. Calvert, J., Mao, Q., Rogers, A. J., Barton, C., Jay, M., Desautels, T., Mohamdlou, H., Jan, J., & Das, R. (2016). A computational approach to mortality prediction of alcohol use disorder inpatients. *Computers in biology and medicine*, Vol. 75, pp. 74–79.
6. Choi, E., Bahadori, M. T., Schuetz, A., Stewart, W. F., & Sun, J. (2016). Doctor ai: Predicting clinical events via recurrent neural networks. *Machine Learning for Healthcare Conference*, pp. 301–318.
7. Crammer, K., Dredze, M., Ganchev, K., Talukdar, P. P., & Carroll, S. (2007). Automatic code assignment to medical text. *Proceedings of the workshop on bionlp 2007: Biological, translational, and clinical language processing*, Association for Computational Linguistics, pp. 129–136.

8. Harutyunyan, H., Khachatrian, H., Kale, D. C., & Galstyan, A. (2017). Multitask learning and benchmarking with clinical time series data. *arXiv preprint arXiv:1703.07771*.
9. Himes, B. E., Dai, Y., Kohane, I. S., Weiss, S. T., & Ramoni, M. F. (2009). Prediction of chronic obstructive pulmonary disease (COPD) in asthma patients using electronic medical records. *Journal of the American Medical Informatics Association*, Vol. 16, No. 3, pp. 371–379.
10. Jin, Z., Sun, Y., & Cheng, A. C. (2009). Predicting cardiovascular disease from real-time electrocardiographic monitoring: An adaptive machine learning approach on a cell phone. *Engineering in Medicine and Biology Society, 2009. EMBC 2009. Annual International Conference of the IEEE, IEEE*, pp. 6889–6892.
11. Johnson, A. E., Pollard, T. J., Shen, L., Li-wei, H. L., Feng, M., Ghassemi, M., Moody, B., Szolovits, P., Celi, L. A., & Mark, R. G. (2016). MIMIC-III, a freely accessible critical care database. *Scientific data*, Vol. 3, pp. 160035.
12. Jonquet, C., Shah, N. H., & Musen, M. A. (2009). The open biomedical annotator. *Summit on translational bioinformatics*, Vol. 2009, pp. 56.
13. Kansagara, D., Englander, H., Salanitro, A., Kagen, D., Theobald, C., Freeman, M., & Kripalani, S. (2011). Risk prediction models for hospital readmission: a systematic review. *Jama*, Vol. 306, No. 15, pp. 1688–1698.
14. Mikolov, T., Chen, K., Corrado, G., & Dean, J. (2013). Efficient estimation of word representations in vector space. *arXiv preprint arXiv:1301.3781*.
15. Miotto, R., Li, L., Kidd, B. A., & Dudley, J. T. (2016). Deep patient: an unsupervised representation to predict the future of patients from the electronic health records. *Scientific reports*, Vol. 6, pp. 26094.
16. Pirracchio, R., Petersen, M. L., Carone, M., Rigon, M. R., Chevret, S., & van der Laan, M. J. (2015). Mortality prediction in intensive care units with the Super ICU Learner Algorithm (SICULA): a population-based study. *The Lancet Respiratory Medicine*, Vol. 3, No. 1, pp. 42–52.
17. Purushotham, S., Meng, C., Che, Z., & Liu, Y. (2018). Benchmarking deep learning models on large healthcare datasets. *Journal of biomedical informatics*.
18. Snomed, C. (2011). Systematized nomenclature of medicine-clinical terms. *International Health Terminology Standards Development Organisation*.
19. Van Houdenhoven, M., Nguyen, D.-T., Eijkemans, M. J., Steyerberg, E. W., Tilanus, H. W., Gommers, D., Wullink, G., Bakker, J., & Kazemier, G. (2007). Optimizing intensive care capacity using individual length-of-stay prediction models. *Critical Care*, Vol. 11, No. 2, pp. R42.
20. Vijayarani, S. & Dhayanand, S. (2015). Data mining classification algorithms for kidney disease prediction. *International Journal on Cybernetics and Informatics (IJCI)*.
21. Xie, P. & Xing, E. (2018). A neural architecture for automated ICD coding. *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, volume 1, pp. 1066–1076.

Article received on 02/02/2019; accepted on 04/03/2019.  
Corresponding author is Gokul S. Krishnan.

# Promoting the Knowledge of Source Syntax in Transformer NMT

Thuong-Hai Pham, Dominik Macháček, Ondřej Bojar

Charles University,  
Faculty of Mathematics and Physics,  
Institute of Formal and Applied Linguistics, Prague,  
Czech Republic

{pham,machacek,bojar}@ufal.mff.cuni.cz

**Abstract.** The utility of linguistic annotation in neural machine translation has been already established. The experiments were however limited to recurrent sequence-to-sequence architectures and relatively small data settings. We focus on the state-of-the-art Transformer model and use comparably larger corpora. Specifically, we try to promote the knowledge of source-side syntax using multi-task learning either through simple data manipulation techniques or through a dedicated model component. The novel idea is to interpret Transformer self-attention as a dependency parse. While the data manipulation techniques are ineffective in large data settings, the treatment of self-attention as dependencies helps in translation and reveals that Transformer model can very easily grasp this structure.

**Keywords.** Syntax, transformer NMT, multi-task NMT.

## 1 Introduction

Neural machine translation (NMT) has dominated the field of MT and many works are emerging that document that the quality of NMT can be, under some circumstances, further improved by incorporating linguistic information from the source and/or target side. Experiments so far were however limited to the recurrent sequence-to-sequence architectures [5, 2].

The latest WMT evaluation [4, 24] show that the novel Transformer architecture [36] has set the new benchmark and it is thus interesting to see if providing this architecture with linguistic information is equally helpful or if Transformer already models the phenomena unsupervised.

We experiment with German-to-Czech and Czech-to-English translation and focus on source-side dependency annotation using multi-task techniques. We try two ways of forcing the model to consider source syntax: (1) by linearizing the syntactic tree and mixing the translation and parsing training examples, and (2) by adding a secondary objective to interpret one of the attention heads as the syntactic tree.

In Section 2, we survey recent experiments with incorporating linguistic information into NMT, focusing particularly on works which use multi-task learning strategies and on works that consider the syntactic analysis of the sentence. A brief description of the data and common settings of our experiments is provided in Section 3. In Section 4, we explore the simple technique of multi-task by alternating training examples of the individual tasks. The main positive contribution of this work is presented in Section 5, where we interpret the self-attention matrix in the Transformer architecture as the dependency tree of the source sentence. Section 6 discusses the observations and we conclude in Section 7.

## 2 Related Work

The idea of multi-task training is to benefit from inherent and implicit similarities between two or more machine learning tasks. If the tasks are solved by a joint model with fewer or more parameters shared among the tasks, the model should exploit the commonalities and perform better in one or more tasks.

This improvement can come from various sources, including the additional (often different) training data used in the additional tasks or some form of regularization or generalization that the other tasks promote.

In machine translation, multitasking has brought interesting results in multi-lingual MT systems and also in using additional linguistic annotation [18, 40, 9, 11].

[8] combined translation and dependency parsing by sharing the translation encoder hidden states with the buffer hidden states in a shift-reduce parsing model [7]. Aiming at the same goal, [1] proposed a very simple method. Instead of modifying the model structure, they represented the target sentence as a linearized lexicalized constituency tree. Subsequently, a sequence-to-sequence (seq2seq) model [32] was used to translate the source sentence to this linearized tree, i.e. indeed performing the two tasks: producing the string of the target sentence jointly with its syntactic analysis. [17] use the same trick for (target-side) dependency trees, proposing a tree-traversal algorithm to linearize the dependency tree. Unfortunately, their algorithm was limited to projective trees.

In parallel to our work, [12] examined various scheduling strategies for a very simple approach to multi-tasking: representing all the tasks converted to a common format of source and target sequences of symbols from a joint vocabulary and training one sequence-to-sequence system on the mix of training examples from the different tasks. The scheduling strategy specified the proportion of the tasks in training batches in time. [12] report improvements in BLEU score [23] in small-data setting for German-to-English translation for all multi-task setups (translation combined with POS tagging and/or source-side<sup>1</sup> parsing). In the “standard” data size and the opposite translation direction, results are mixed and only one of the scheduling strategies and only the POS secondary task help to improve MT over the baseline.

<sup>1</sup>[12] do not explicitly state whether they use the source or the target language treebank as the training data for the parsing task. While both is actually possible, and while even the combination of both could be tried, we assume they used the source-side treebank only.

The papers mentioned so far targeted primarily the quality of MT (as measured by BLEU), not the secondary tasks. [12] note that their system performs reasonably well in both tagging and parsing. [28] present an in-depth analysis of the syntactic knowledge learned by the recurrent sequence-to-sequence NMT. [35] are the first to use Transformer and observe that the recurrence is indeed important to model hierarchical structures.

[19] benefit from CCG tags [30] added to NMT on the source side in the form of word factors and on the target side by interleaving the CCG tags and target words. The additional information proves useful when the CCG tags and words are processed in sync. [33] report similar success in interleaving words and morphological tags.

### 3 Data and Common Settings

Experiments in this paper are based on two language pairs: German-to-Czech (de2cs) translation trained on Europarl [15] and OpenSubtitles2016 [34], after some cleanup preprocessing, character normalization and tokenization. These are the only publicly available parallel data for this language pair. Czech-to-English (cs2en) translation was trained on a subset of CzEng 1.7 [3]<sup>2</sup>. The data sizes used for MT training are summarized in Table 1. For training of parsing tasks, we used the same datasets automatically annotated on source sides. For German source we used UDPipe [31], with the model trained on Universal Dependencies 2.0 (UD, [21]). For Czech source we used the annotation provided in CzEng release, originally created by Treex [26]. This annotation is based on Prague Dependency Treebank (PDT, [10]). For parsing evaluation we used gold test set from UD and PDT, respectively.

We use several automatic evaluation metrics to assess translation quality: BLEU [23], CharacTER [37], BEER [29], and chrF3 [27]. For experiments in Section 4, the BLEU score is cased, implemented within T2T,<sup>3</sup> in Section 5 with sacrebleu<sup>4</sup>.

<sup>2</sup><http://ufal.mff.cuni.cz/czeng>

<sup>3</sup>[https://github.com/tensorflow/tensor2tensor/blob/master/tensor2tensor/utils/bleu\\_hook.py](https://github.com/tensorflow/tensor2tensor/blob/master/tensor2tensor/utils/bleu_hook.py)

<sup>4</sup><https://github.com/aws-labs/sockeye/tree/master/contrib/sacrebleu>

**Table 1.** Data used in our experiments. Test and dev data for de2cs originate in WMT newstests, all data for cs2en in indicated blocks of CzEng.

Dataset	de2cs	cs2en
Train sent. pairs	8.8M	#00-#08: 5.2M
Train tokens (src/tgt)	89M/78M	61M/69M
Test sent. pairs	news 2013: 3k	#09: 10k
Dev sent. pairs	news 2011: 3k	#09: 1k

For dependency parsing task, we use unlabeled attachment score (UAS).

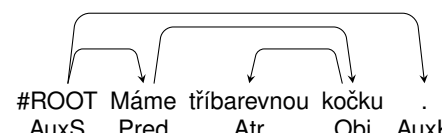
To assess the significance of the improvement over a given baseline, we use MT-ComparEval [13], which implemented the paired bootstrap resampling test (confidence level 0.05 or 0.01; [14]).

## 4 Simple Alternating Multi-Task

### 4.1 Approach

For simple multi-task learning, where the input and output of each task are represented as sequences, the basic architecture for MT can be used without any modifications. The identification of the task can be provided by a special token on the source side, which we add as the very last symbol of the sentence. The encoder and decoder of the NMT model are thus shared for all tasks which enables the encoder to learn source language better, but on the other hand it occupies a certain part of the model with task alternation and multiple language models for each task in the decoder.

In our experiments, we mix two tasks: MT and one additional linguistic (see Section 4.1) or dummy referential task (see Section 2). In “DepHeads” task, word forms of nodes’ parents in the dependency tree are predicted. We can reconstruct unlabeled dependency tree in a post-processing step. If one word form appears multiple times in a sentence, we attach the edge to the nearest option. We propose this approach mostly for annotation schemes, in which content words (in contrast to function words) appear as inner nodes of dependency trees. Since content words are usually not repeated in sentences, there is a low chance they will be mismatched.



#ROOT	Máme	tříbarevnou	kočku	.
AuxS	Pred	Atr	Obj	AuxK

MT source (cs)	Máme tříbarevnou kočku . #Translate	
MT target (en)	We have a three-colored cat .	
DepHeads	src	Máme tříbarevnou kočku . #DepHeads
	tgt	#ROOT kočku Máme #ROOT
DepLabels	src	Máme tříbarevnou kočku . #DepLabels
	tgt	#Pred #Atr #Obj #AuxK
DepHeads	src	Máme tříbarevnou kočku . #DHeadsLab
+DepLabels	tgt	#ROOT #Pred kočku #Atr Máme #Obj #ROOT #AuxK

**Fig. 1.** Sample dependency tree, inputs and expected outputs of linguistic secondary tasks

Source words	We have a three-colored cat .
CountSrcWords	6
EnumSrcWords	W W W W W W
CopySrc	We have a three-colored cat .

**Fig. 2.** Sample inputs and expected outputs of dummy secondary tasks

“DepLabels” task is tagging with dependency labels, and “DepHeads+DepLabels” is an interleaved combination of the two.

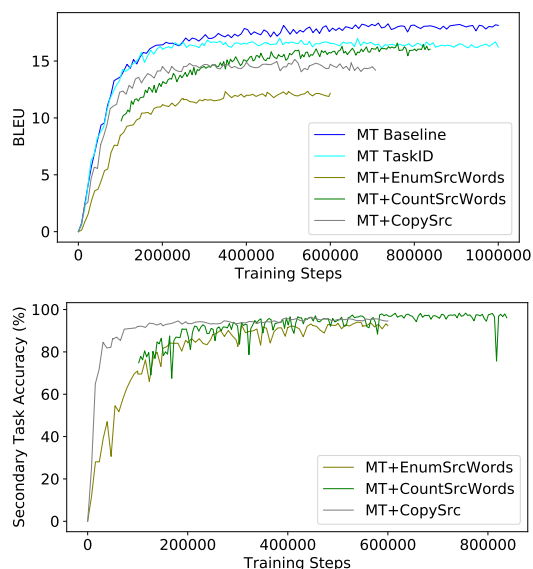
The training data in multitasking are selected by constant scheduler as in [12], with parameter 0.5, which means the trainer alternates between the tasks, in average, after every training step. As [12] reminds, this is different from [18, 39] where the mixing happens at the level of batches and not individual examples.

The experiments here in Section 4 used Google’s Tensor2Tensor version 1.2.9, transformer.big\_single\_gpu hyperparameter set (hidden size 1024, filter size 4096, 16 self-attention heads, 6 layers) with batch size 1500, 60k warmup steps and 100k shared vocabulary provided by T2T’s default SubwordTextEncoder.

### 4.2 Training Cost of the Multi-Task

Adding training examples of the secondary task is bound to affect the training throughput and speed.<sup>5</sup>

<sup>5</sup>We adopt the terminology of [25].

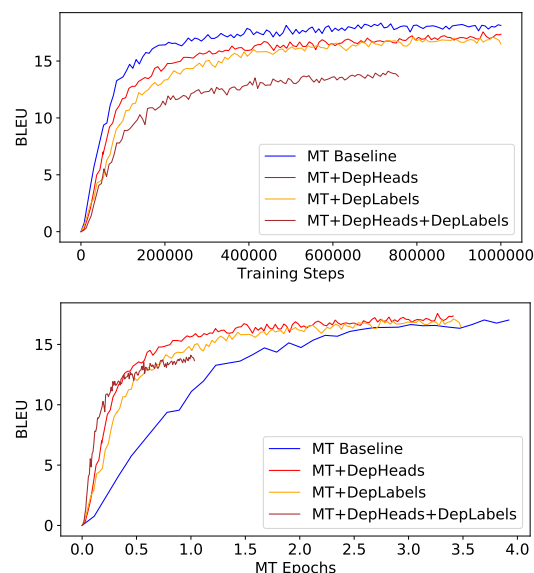


**Fig. 3.** Learning curves of the de2cs baseline and dummy secondary tasks over training steps. MT BLEU on top, percentage of correct answers for secondary task on bottom

The hope is that this extra training cost is worth the gains obtained in the main task. We examine it empirically by comparing the training speed of the baseline run (no multi-task) and several versions of “dummy” multi-task setups as illustrated in Figure 2. In “CountSrcWords”, the system is expected to count the source words and emit the result as one token holding the decimal number. “EnumSrcWords” is similar but the expected output is much easier for the architecture to grasp: the count should be expressed by an appropriate number of copies of the same special output token. In “CopySrc”, the system should simply learn to copy the source, which should be very easy for an attentive architecture.

The task identification is clearly marked on input with a special token. To measure its impact on MT quality, we provide an experimental run “MT TaskID”, where only one MT task with task identification token is provided.

Figure 3 summarizes the resulting learning curves on the development set. As we supposed, the secondary dummy task was easy to learn but it



**Fig. 4.** Learning MT BLEU curves of the de2cs baseline and linguistic secondary tasks over training steps (top) and over MT epochs (bottom)

hurts MT performance. Enumerating and counting full words are very similar tasks in difficulty, the model learned them almost in same time, but enumerating worsens MT quality much more. It probably employs bigger part of decoder. A surprising result is that the task identification token on baseline MT data decreases overall MT performance in the long run.

#### 4.3 Results of Simple Alternating Multi-Task

As Table 2 and Figure 4 (top) indicate, none of simple alternating multi-tasking method with linguistic secondary task outperformed baseline MT on any of our language pairs after the same amount of time. (In our conditions, training steps and training time are easily convertible; 600k training steps correspond to approximately 40 hours).

However, if we measure the performance on MT training data throughput, we see the multi-tasking runs achieved the same level as the baseline with less training data. We conclude that the cost for sharing encoder and decoder between two tasks



**Table 2.** Automatic scores for MT with multi-task by simple alternation. All experiments are after 600k of training steps. Scores: BLEU, Character, BEER, and chrF3. Best in bold, second-best slanted. Statistical significance marked as † ( $p < 0.05$ ) and ‡ ( $p < 0.01$ ) when compared to the second-best

Model	de2cs								cs2en							
	dev				test				dev				test			
MT Baseline	<b>17.90</b> ‡	<b>60.73</b>	<b>52.30</b>	<b>47.06</b>	<b>19.74</b> ‡	<b>58.60</b>	<b>53.08</b>	<b>48.62</b>	<b>44.92</b>	<b>42.11</b>	<b>63.32</b>	<b>65.34</b>	<b>44.20</b>	<b>41.68</b>	<b>62.70</b>	<b>63.88</b>
MT+DepLabels	16.52	62.67	50.86	45.18	17.87	59.65	51.67	47.01	41.98	43.28	61.80	63.63	41.94	42.54	61.63	61.96
MT+DepHeads	16.36	62.55	50.76	45.21	17.51	62.15	51.29	46.52	40.72	43.75	61.85	62.78	41.10	42.30	61.41	61.68
MT+DepHeads+DepLabels	13.62	70.25	48.52	43.06	15.45	67.14	49.69	44.79	39.57	45.63	60.50	61.30	40.25	43.63	60.97	61.05

**Table 3.** Comparison of BLEU scores at 600k training steps for linguistic and dummy secondary tasks with simple alternating approach

Model	dev	test
MT Baseline	<b>17.90</b>	<b>19.74</b>
MT TaskID	16.53	18.20
MT+DepLabels	16.52	17.87
MT+DepDheads	16.36	17.62
MT+CountSrcWords	15.70	17.51
MT+CopySrc	14.73	16.07
MT+DepHeads+DepLabels	13.62	15.45
MT+EnumSrcWords	12.16	14.04

is higher than benefits from additional linguistic resources, but in particularly small data settings multi-tasking may be desirable.

Table 3 shows comparison between linguistic and dummy secondary tasks. “DepHeads” and “DepLabels” outperformed “CountSrcWords” and all other dummy tasks, so we conclude the model gains from semi-supervised syntactic input.

Table 4 shows the performance in parsing. As the referential parser, we use UDPipe for German, the one which supervised our model. Our system gains a similar UAS performance. It should be noted that we used the supervision by UDPipe in a non-standard way. Our system (and the referential parser) take raw word tokens on input, while UDPipe is designed to segment multi-word tokens, such as the German *zum*, into syntactic words, as *zu dem*, each of which are single nodes in tree.

For Czech, we report the score of winner in CoNLL Shared Task 2007 [22], the latest available evaluation on same data. We expect that the state of the art is higher nowadays. The limitation of our

**Table 4.** Test set scores for parsing source language (German and Czech, resp.) by simple alternation. “label acc” is the accuracy of tagging words with their dependency labels. Best in bold, second-best slanted

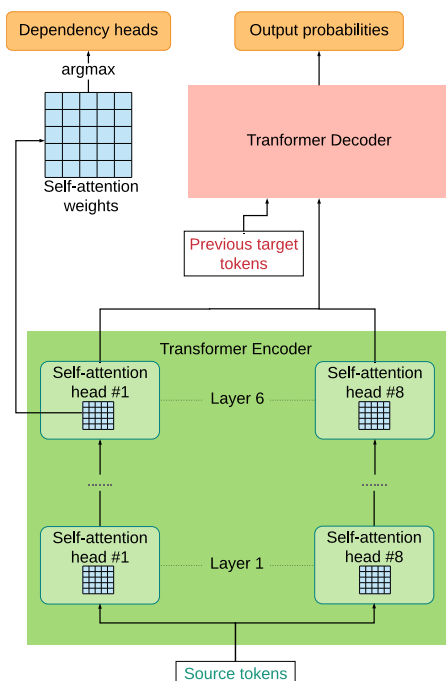
Model	de2cs		cs2en	
	UAS	label acc	UAS	label acc
referential parser	<b>62.87</b>	73.62	<b>86.28</b>	83.38
MT+DepLabels	–	<b>75.40</b>	–	<b>85.01</b>
MT+DepHeads	62.15	–	80.35	–
MT+DepHeads+DepLabels	54.98	68.44	80.01	83.99

model may be the shared decoder and potentially inaccurate automatically annotated training data.

## 5 Promoting Dependency Interpretation of Self-Attention

In this section, we propose a different but similarly simple technique to promote explicit knowledge of source syntax in the model. Our inspiration comes from the neural model for dependency parsing by [6]. The model produces a matrix  $S(u, v)$  expressing the probability that the word  $u$  is the head of  $v$ . The construction of this matrix is very similar to the matrix of self-attention weights  $\alpha$  in the Transformer model.

From this similarity, we speculate that the self-attentive architecture of Transformer NMT has the capacity to learn dependency parsing and we only need to promote a little the particular linguistic dependencies captured in a treebank.



**Fig. 5.** Joint dependency parsing and translation model (“DepParse”)

### 5.1 Model Architecture

Figure 5 illustrates our joint model (“DepParse”). The translation part is kept unchanged. The only difference is that we reinterpret one of the self-attention heads in the Transformer encoder as if it was the dependency matrix  $S(u, v)$ . The training objective is combined and maximizes both the translation quality in terms of cross-entropy of the candidate translation and the unlabeled attachment score (UAS) of the proposed head against the (automatic) golden parse. The particular choice of the head which will serve as the dependency parser is arbitrary. Put differently, we constrain the Transformer model to use one of its heads to follow the given syntactic structure of the sentence.

It would be also possible to use e.g. the deep-syntactic parse of the sentence (the tectogrammatical layer as defined e.g. for the Prague Dependency Treebank, [10]); we leave that for future work.

**Table 5.** DepParse’s results in translation (BLEU) and parsing (UAS) on automatically annotated (cs2en). All test BLEU gains, except for layer 0, are statistically significant with  $p < 0.01$  when compared to TransformerBase.

	BLEU		UAS	
	Dev	Test	Dev	Test
TransformerBase	37.28	36.66	–	–
Parse from layer 0	36.95	36.60	81.39	82.85
Parse from layer 1	<b>38.51</b>	<b>38.01</b>	90.17	90.78
Parse from layer 2	38.50	37.87	91.31	91.18
Parse from layer 3	38.37	37.67	91.43	91.43
Parse from layer 4	37.86	37.60	<b>91.65</b>	<b>91.56</b>
Parse from layer 5	37.63	37.67	91.44	91.46

### 5.2 Experiment Setup

Experiments in this section were carried out with T2T version 1.5.6 at the *word level*, i.e. without using subword units. We decided for this simplification for an easier alignment between the translation and parsing tasks.

The Transformer hyper-parameter set `transformer_base` [25] was used for all model variants with hidden size 512, filter size 2048, 8 self-attention heads and 6 layers in each of the encoder and decoder. From now on, we refer the Transformer model with this hyper-parameter set as “TransformerBase”, our baseline. We also experimented with the choice of the encoder layer, which we use for parsing.

In addition to the standard preprocessing for MT, we inserted a special “ROOT” word to the beginning of every sentence, so that the selected self-attention head would be able to represent a dependency tree correctly.

### 5.3 Layer Choice

Firstly, we experiment with selection of one of the six encoder layers from which we take the self-attention head that will serve as the dependency parse. Table 5 presents the results for both translation and parsing.

It is apparent that layer 0 (the first layer) is a too early stage for both tasks. The self-attention mechanism has only access to input word embeddings, and their relations are very likely

**Table 6.** BLEU scores on test set for translation task (T2T 1.5.6, word level). Statistical significance marked as † ( $p < 0.05$ ) and ‡ ( $p < 0.01$ ) when compared to TransformerBase

Model	de2cs	cs2en
TransformerBase	13.96	36.66
Alternating multi-tasking (Section 4)	12.85	36.47
DepParse (Section 5)	<b>14.27<sup>†</sup></b>	<b>38.01<sup>‡</sup></b>

**Table 7.** UAS on gold annotated test sets for parsing task

Model	de2cs	cs2en
Referential parsers	62.87	86.28
DepParse	76.48	82.53

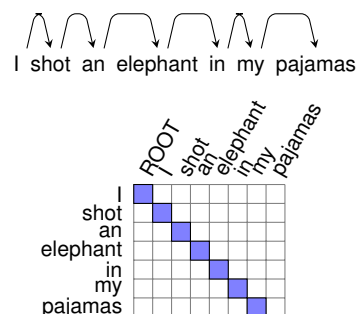
to be useful semantically rather than syntactically. On the other hand, layers 1 and 2 perform well in parsing, and they are the best layers for translation quality. A possible explanation is that they already have sufficient information for a reasonably precise parse and do not consume the encoder's capacity for translation. Further layers perform generally better and better in parsing (because they are more informed) and maintain a solid performance in translation, but the translation quality is slowly decreasing. For the following, we select layer 1 to demand syntactic information from.

## 5.4 Performance in Translation

Table 6 compares the performance of the baseline Transformer, the simple alternating setup from Section 4 (DepHeads src) and the multi-task setup from this section. All these runs use T2T version 1.5.6 and use words, not subword units. This also explains the decrease in BLEU compared to Table 2. The DepParse model significantly outperforms the baseline (38.01 vs. 36.66 and 14.27 vs. 13.96).

## 5.5 Performance in Parsing

In addition to the automatically annotated dev and test set, we also evaluated our model on the gold evaluation sets from UD 2.0 for German and from PDT 2.5 for Czech. The referential parsers



**Fig. 6.** Dummy dependencies with diagonal matrix (the columns represent the heads, the rows are dependents)

were defined in Table 4. Table 7 shows that our model achieved good results in comparison to the baseline model on those datasets, even though ours was trained using synthetic data.

## 5.6 Diagonal Parse

For contrast, we conduct an experiment with a simpler sentence structure, which we call the “diagonal parse”. In the diagonal parse, the dependency head of a token is simply the previous token, as illustrated in Figure 6.

Our model for the joint diagonal parsing and translation (“DiagonalParse”) is identical to the “DepParse” model, which has been described in Section 5.1. We only use diagonal matrices during training, instead of the dependency matrices. The main goal of this setup is to examine the benefits of the additional syntactic information for machine translation.

Table 8 documents that the “DiagonalParse” model is very effective. The diagonal parsing precision is, as expected, very high, ranging from 99.95% to 99.99% on the test set. This joint model also outperformed the baseline in translation task with all its variants (BLEU scores vary from 37.47 to 38.14, compared to 36.66). Moreover, these results form an observable pattern, in which the best result comes from the model with parsing with the head on layer 0. Parsing from deeper layers still helps to improve translation over baseline, but the BLEU scores decrease. We believe that a possible explanation for this pattern is that the diagonal matrix represents the

**Table 8.** DiagonalParse's results in translation (BLEU) and diagonal parsing (precision) on cs2en. All test BLEU improvements are statistically significant with  $p < 0.01$  when compared to the TransformerBase

	BLEU		Precision	
	Dev	Test	Dev	Test
TransformerBase	37.28	36.66	—	—
Parse from layer 0	38.68	<b>38.14</b>	99.97	99.96
Parse from layer 1	<b>39.11</b>	38.06	<b>99.99</b>	<b>99.99</b>
Parse from layer 2	37.85	37.85	99.98	99.98
Parse from layer 3	37.93	37.70	99.97	99.98
Parse from layer 4	37.68	37.47	99.98	99.96
Parse from layer 5	37.53	37.54	99.96	99.95

relation between the preceding token and the current token. This simple sentence structure can serve as an additional positional information to the absolute positional embeddings. Therefore, the sooner the model is forced to recognize this positional information (via training the parsing task), the better it can learn to translate. Another possible explanation is the regularization effect of the diagonal parse.

### 5.7 Training Speed

While having achieved the results discussed above, the training costs for our multi-task models are comparable to the baseline Transformer. The training time (including internal evaluation every 1000 steps) on a single GPU NVIDIA GTX 1080 Ti needed to reach 250k steps for TransformerBase was about 1 day and 4 hours while our joint models needed only 10%-13% more time to train on both tasks.

### 5.8 Self-Attention Patterns in the Encoder

Figure 7 presents the behavior of self-attention mechanism in each layer of our models for the first 100 sentences in the test set. The bin  $[0.0, 0.1)$  was excluded from the picture for clarity because most of the self-attention weights fall into this trivial bin. As can be seen from the figure, the layers in which the model was trained to parse display a very sharp attention, i.e. for each head, each word attends to only one or two words from the previous layer. This

behavior is apparent in all our multi-task models except the “Parse from layer 0”. As mentioned in Section 5.4, this model performed badly on both tasks. While the causality is unclear, we at least see that the sharpness in attention is related to the better performance.

Figure 8 documents another interesting observation (as above, the bin  $[0.0, 0.1)$  was excluded). One could perhaps expect that the particular head trained to predict dependencies will have a sharp attention but interestingly, the same sharpness is observed in all heads of the given layer. A possible reason may be due to the vector concatenation and layer normalization after each multi-head attention layer in the Transformer.

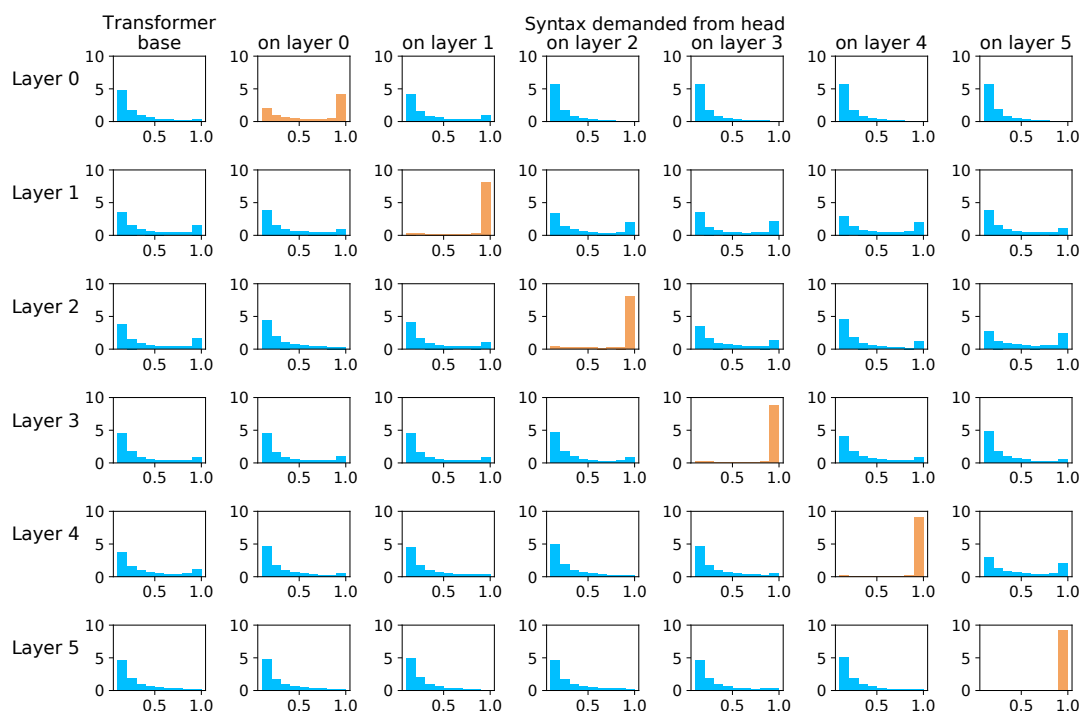
## 6 Discussion

[12] suggest another representation of “Dep-Heads”, which doesn't suffer on unknown words and repeated words. They represent head as an offset from the node's position represented as decimal number, positive to the right, negative to the left. We showed Transformer can easily learn to count words. This representation should be considered in future work.

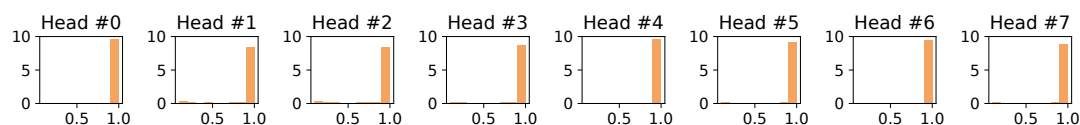
We let aside a question of vocabulary design for multi-tasking. In T2T's SubwordTextEncoder (STE), the vocabulary is designed unsupervisedly from a training data sample, so that frequent words are represented as single subwords and rare words as sequences of characters. We assume that advantaging different sides of particular tasks on STE input can lead to better quality. This could be combined with different parameters for the constant task scheduler.

Multiple multi-task experiments ([20], [38], etc.) mention notable gains on small data scenarios. As documented by [16], under certain training data size, NMT is actually much worse than conventional phrase-based MT. It is unclear if the gains from NMT multi-tasking are obtained also after this critical corpus size, or if they are limited to the data sizes where NMT is ineffective.

One limitation of our setup was that our model was trained on automatic parses. Hence, it would be interesting to fine-tune our model with



**Fig. 7.** Histogram of normalized self-attention weights in the encoder



**Fig. 8.** Histogram of self-attention weights in the encoder's layer 4 when parsing from layer 4

gold-annotated trees, which could lead to a better parsing performance. We leave this for future work.

## 7 Conclusion

We proposed two techniques of promoting the knowledge of source syntax in the Transformer model of NMT by multi-tasking and evaluated them at reasonably large data sizes.

The simple data manipulation technique, alternating translation and linearized parsing, is impractical. Learning to translate and parse improves over comparable multi-task setups with uninformative (“dummy”) secondary tasks, but overall it performs worse than single-task

translation model. In low-resource conditions, the gain from the multi-tasking may be useful.

The other technique, re-interpreting one of the self-attention heads in the Transformer model as the dependency analysis of the sentence, is surprisingly effective. At little or no cost in training time, Transformer learns to translate and parse at the same time.

The parse accuracy is reasonable and the translation is significantly better than the baseline. Curiously, very similar gains can be obtained by predicting a “diagonal parse”, i.e. linguistically uninformed linear tree. The full explanation of this behavior is yet to be sought for.

## Acknowledgments

The research was partially supported by the grants 19-26934X (NEUREM3) of the Czech Science Foundation and H2020-ICT-2018-2-825460 (ELITR) of the EU.

## References

1. Aharoni, R. & Goldberg, Y. (2017). Towards string-to-tree neural machine translation. *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics, ACL 2017, Vancouver, Canada, July 30 - August 4, Volume 2: Short Papers*, pp. 132–140.
2. Bahdanau, D., Cho, K., & Bengio, Y. (2015). Neural machine translation by jointly learning to align and translate. *Proceedings of ICLR*.
3. Bojar, O., Dušek, O., Kocmi, T., Libovický, J., Novák, M., Popel, M., Sudarikov, R., & Variš, D. (2016). CzEng 1.6: Enlarged Czech-English Parallel Corpus with Processing Tools Dockered. Sojka, P., Horák, A., Kopeček, I., & Pala, K., editors, *Text, Speech, and Dialogue: 19th International Conference, TSD 2016*, number 9924 in Lecture Notes in Computer Science, Masaryk University, Springer International Publishing, Cham / Heidelberg / New York / Dordrecht / London, pp. 231–238.
4. Bojar, O., Federmann, C., Fishel, M., Graham, Y., Haddow, B., Huck, M., Koehn, P., & Monz, C. (2018). Findings of the 2018 conference on machine translation (wmt18). *Proceedings of the Third Conference on Machine Translation, Volume 2: Shared Task Papers*, Association for Computational Linguistics, Belgium, Brussels, pp. 272–307.
5. Cho, K., van Merriënboer, B., Gulcehre, C., Bahdanau, D., Bougares, F., Schwenk, H., & Bengio, Y. (2014). Learning phrase representations using rnn encoder–decoder for statistical machine translation. *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, Association for Computational Linguistics, Doha, Qatar, pp. 1724–1734.
6. Dozat, T. & Manning, C. D. (2016). Deep biaffine attention for neural dependency parsing. *CoRR*, Vol. abs/1611.01734.
7. Dyer, C., Kuncoro, A., Ballesteros, M., & Smith, N. A. (2016). Recurrent neural network grammars. *HLT-NAACL, The Association for Computational Linguistics*, pp. 199–209.
8. Eriguchi, A., Tsuruoka, Y., & Cho, K. (2017). Learning to parse and translate improves neural machine translation. *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics, ACL 2017, Vancouver, Canada, July 30 - August 4, Volume 2: Short Papers*, pp. 72–78.
9. Ha, T.-L., Niehues, J., & Waibel, A. (2016). Toward Multilingual Neural Machine Translation with Universal Encoder and Decoder. *Proceedings of the International Workshop on Spoken Language Translation, IWSLT'16*, Seattle, USA.
10. Hajič, J., Panevová, J., Hajičová, E., Sgall, P., Pajas, P., Štěpánek, J., Havelka, J., Mikulová, M., Žabokrtský, Z., & Ševčíková Razimová, M. (2006). Prague Dependency Treebank 2.0. LDC2006T01, ISBN: 1-58563-370-4.
11. Johnson, M., Schuster, M., Le, Q. V., Krikun, M., Wu, Y., Chen, Z., Thorat, N., Viégas, F. B., Wattenberg, M., Corrado, G., Hughes, M., & Dean, J. (2016). Google's multilingual neural machine translation system: Enabling zero-shot translation. *CoRR*, Vol. abs/1611.04558.
12. Kiperwasser, E. & Ballesteros, M. (2018). Scheduled Multi-Task Learning: From Syntax to Translation. *Transactions of the Association for Computational Linguistics*, Vol. 6, pp. 225–240.
13. Kleijch, O., Avramidis, E., Burchardt, A., & Popel, M. (2015). Mt-compareval: Graphical evaluation interface for machine translation development. *The Prague Bulletin of Mathematical Linguistics*, Vol. 104, No. 1, pp. 63–74.
14. Koehn, P. (2004). Statistical significance tests for machine translation evaluation. *Proceedings of EMNLP*, volume 4, pp. 388–395.
15. Koehn, P. (2005). Europarl: A Parallel Corpus for Statistical Machine Translation. *Conference Proceedings: the tenth Machine Translation Summit*, AAMT, AAMT, Phuket, Thailand, pp. 79–86.
16. Koehn, P. & Knowles, R. (2017). Six challenges for neural machine translation. *Proceedings of the First Workshop on Neural Machine Translation*, Association for Computational Linguistics, Vancouver, pp. 28–39.
17. Le, A. N., Martinez, A., Yoshimoto, A., & Matsumoto, Y. (2017). Improving sequence to

- sequence neural machine translation by utilizing syntactic dependency information. *Proceedings of the Eighth International Joint Conference on Natural Language Processing, IJCNLP 2017, Taipei, Taiwan, November 27 - December 1, 2017 - Volume 1: Long Papers*, pp. 21–29.
18. Luong, M., Le, Q. V., Sutskever, I., Vinyals, O., & Kaiser, L. (2015). Multi-task sequence to sequence learning. *CoRR*, Vol. abs/1511.06114.
  19. Nadejde, M., Reddy, S., Sennrich, R., Dwojak, T., Junczys-Dowmunt, M., Koehn, P., & Birch, A. (2017). Predicting target language ccg supertags improves neural machine translation. *Proceedings of the Second Conference on Machine Translation, Volume 1: Research Paper*, Association for Computational Linguistics, Copenhagen, Denmark, pp. 68–79.
  20. Niehues, J. & Cho, E. (2017). Exploiting linguistic resources for neural machine translation using multi-task learning. *WMT*.
  21. Nivre, J., Agić, Ž., Ahrenberg, L., Antonsen, L., Aranzabe, M. J., Asahara, M., Ateyah, L., Attia, M., Atutxa, A., Badmaeva, E., Ballesteros, M., Banerjee, E., Bank, S., Bauer, J., Bengoetxea, K., Bhat, R. A., Bick, E., Bosco, C., Bouma, G., Bowman, S., Burchardt, A., Candito, M., Caron, G., Cebiroğlu Eryiğit, G., Celano, G. G. A., Cetin, S., Chalub, F., Choi, J., Cho, Y., Cinková, S., Çöltekin, Ç., Connor, M., de Marneffe, M.-C., de Paiva, V., Diaz de Ilarraza, A., Dobrovolsky, K., Dozat, T., Droganova, K., Eli, M., Elkahky, A., Erjavec, T., Farkas, R., Fernandez Alcalde, H., Foster, J., Freitas, C., Gajdošová, K., Galbraith, D., Garcia, M., Ginter, F., Goenaga, I., Gojenola, K., Gökırmak, M., Goldberg, Y., Gómez Guinovart, X., Gonzáles Saavedra, B., Grioni, M., Grūzītis, N., Guillaume, B., Habash, N., Hajič, J., Hajič jr., J., Hà Mỹ, L., Harris, K., Haug, D., Hladká, B., Hlaváčová, J., Hohle, P., Ion, R., Irimia, E., Johannsen, A., Jørgensen, F., Kaşıkara, H., Kanayama, H., Kanerva, J., Kayadelen, T., Kettnerová, V., Kirchner, J., Kotsyba, N., Krek, S., Kwak, S., Laippala, V., Lambertino, L., Lando, T., Lê H'ông, P., Lenci, A., Lertpradit, S., Leung, H., Li, C. Y., Li, J., Ljubešić, N., Loginova, O., Lyashevskaya, O., Lynn, T., Macketanz, V., Makazhanov, A., Mandl, M., Manning, C., Manurung, R., Măranduc, C., Mareček, D., Marheinecke, K., Martínez Alonso, H., Martins, A., Mašek, J., Matsumoto, Y., McDonald, R., Mendonça, G., Missilä, A., Mititelu, V., Miyao, Y., Montemagni, S., More, A., Moreno Romero, L., Mori, S., Moskalevskyi, B., Muischnek, K., Mustafina, N., Müürisep, K., Nainwani, P., Nedoluzhko, A., Nguyễn Thị, L., Nguyễn Thị Minh, H., Nikolaev, V., Nitisaroj, R., Nurmi, H., Ojala, S., Osenova, P., Ovelid, L., Pascual, E., Passarotti, M., Perez, C.-A., Perrier, G., Petrov, S., Piitulainen, J., Pitler, E., Plank, B., Popel, M., Pretkalniņa, L., Prokopidis, P., Puolakainen, T., Pyysalo, S., Rademaker, A., Real, L., Reddy, S., Rehm, G., Rinaldi, L., Rituma, L., Rosa, R., Rovati, D., Saleh, S., Sanguinetti, M., Saulite, B., Sawanakunanon, Y., Schuster, S., Seddah, D., Seeker, W., Seraji, M., Shakurova, L., Shen, M., Shimada, A., Shohibussirri, M., Silveira, N., Simi, M., Simionescu, R., Simkó, K., Šimková, M., Simov, K., Smith, A., Stella, A., Strnadová, J., Suhr, A., Sulubacak, U., Szántó, Z., Taji, D., Tanaka, T., Trosterud, T., Trukhina, A., Tsarfaty, R., Tyers, F., Uematsu, S., Urešová, Z., Uria, L., Uszkoreit, H., van Noord, G., Varga, V., Vincze, V., Washington, J. N., Yu, Z., Žabokrtský, Z., Zeman, D., & Zhu, H. (2017). Universal dependencies 2.0. LINDAT/CLARIN digital library at the Institute of Formal and Applied Linguistics (ÚFAL), Faculty of Mathematics and Physics, Charles University.
  22. Nivre, J., Hall, J., Kübler, S., McDonald, R. T., Nilsson, J., Riedel, S., & Yuret, D. (2007). The conll 2007 shared task on dependency parsing. *EMNLP-CoNLL 2007, Proceedings of the 2007 Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning, June 28-30, 2007, Prague, Czech Republic*, pp. 915–932.
  23. Papineni, K., Roukos, S., Ward, T., & Zhu, W.-J. (2002). Bleu: A method for automatic evaluation of machine translation. *Proceedings of the 40th Annual Meeting on Association for Computational Linguistics, ACL '02, Association for Computational Linguistics, Stroudsburg, PA, USA*, pp. 311–318.
  24. Popel, M. (2018). CUNI Transformer Neural MT System for WMT18. *Proceedings of the Third Conference on Machine Translation*, Association for Computational Linguistics, Belgium, Brussels, pp. 486–491.
  25. Popel, M. & Bojar, O. (2018). Training tips for the transformer model. *The Prague Bulletin of Mathematical Linguistics*, Vol. 110, No. 1, pp. 43 – 70.

26. Popel, M. & Žabokrtský, Z. (2010). TectoMT: Modular NLP framework. Loftsson, H., Rognvaldsson, E., & Helgadóttir, S., editors, *Lecture Notes in Artificial Intelligence, Proceedings of the 7th International Conference on Advances in Natural Language Processing (IceTAL 2010)*, volume 6233 of *Lecture Notes in Computer Science*, Iceland Centre for Language Technology (ICLT), Springer, Berlin / Heidelberg, pp. 293–304.
27. Popovic, M. (2015). chrF: character n-gram f-score for automatic MT evaluation. *Proceedings of the Tenth Workshop on Statistical Machine Translation, WMT@EMNLP 2015, 17-18 September 2015, Lisbon, Portugal*, pp. 392–395.
28. Shi, X., Padhi, I., & Knight, K. (2016). Does string-based neural MT learn source syntax? Su, J., Carreras, X., & Duh, K., editors, *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing, EMNLP 2016, Austin, Texas, USA, November 1-4, 2016*, The Association for Computational Linguistics, pp. 1526–1534.
29. Stanojević, M. & Sima'an, K. (2014). Fitting sentence level translation evaluation with many dense features. *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, Association for Computational Linguistics, Doha, Qatar, pp. 202–206.
30. Steedman, M. (2000). *The Syntactic Process*. MIT Press, Cambridge, MA, USA.
31. Straka, M. & Straková, J. (2017). Tokenizing, pos tagging, lemmatizing and parsing ud 2.0 with udpipe. *Proceedings of the CoNLL 2017 Shared Task: Multilingual Parsing from Raw Text to Universal Dependencies*, Association for Computational Linguistics, Vancouver, Canada, pp. 88–99.
32. Sutskever, I., Vinyals, O., & Le, Q. V. (2014). Sequence to sequence learning with neural networks. *NIPS*, pp. 3104–3112.
33. Tamchyna, A., Weller-Di Marco, M., & Fraser, A. (2017). Modeling target-side inflection in neural machine translation. *Proceedings of the Second Conference on Machine Translation, Volume 1: Research Paper*, Association for Computational Linguistics, Copenhagen, Denmark, pp. 32–42.
34. Tiedemann, J. (2009). News from OPUS - A collection of multilingual parallel corpora with tools and interfaces. In *Proc. of RANLP*, volume V. John Benjamins, Amsterdam/Philadelphia, Borovets, Bulgaria, pp. 237–248.
35. Tran, K. M., Bisazza, A., & Monz, C. (2018). The importance of being recurrent for modeling hierarchical structure. *CoRR*, Vol. abs/1803.03585.
36. Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A. N., Kaiser, L., & Polosukhin, I. (2017). Attention is all you need. *Advances in Neural Information Processing Systems 30: Annual Conference on Neural Information Processing Systems 2017, 4-9 December 2017, Long Beach, CA, USA*, pp. 6000–6010.
37. Wang, W., Peter, J.-T., Rosendahl, H., & Ney, H. (2016). Character: Translation edit rate on character level. *Proceedings of the First Conference on Machine Translation*, Association for Computational Linguistics, Berlin, Germany, pp. 505–510.
38. Zareemoodi, P. & Haffari, G. (2018). Neural machine translation for bilingually scarce scenarios: A deep multi-task learning approach.
39. Zoph, B. & Knight, K. (2016). Multi-Source Neural Translation. *Proceedings of the 2016 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, Association for Computational Linguistics, San Diego, California, pp. 30–34.
40. Zoph, B., Yuret, D., May, J., & Knight, K. (2016). Transfer learning for low-resource neural machine translation. *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing*, Association for Computational Linguistics, Austin, Texas, pp. 1568–1575.

Article received on 23/02/2019; accepted on 04/03/2019.  
Corresponding author is Thuong-Hai Pham.



# Reasoning over Arabic WordNet Relations with Neural Tensor Network

Mohamed Ali Batita<sup>1</sup>, Rami Ayadi<sup>2,3</sup>, Mounir Zrigui<sup>1</sup>

<sup>1</sup> University of Monastir, Faculty of Sciences,  
Science Department, Monastir,  
Tunisia

<sup>2</sup> University of Gabes,  
Higher Institute of Computer of Medenine,  
Tunisia

<sup>3</sup> Jouf University, College of Science and Arts in Al Qurayyat,  
Computer Science Department,  
Kingdom of Saudi Arabia (KSA)

batitamohamedali@gmail.com, rayadi@ju.edu.sa, mounir.zrigui@fsm.rnu.tn

**Abstract.** Arabic WordNet is an important resource for many tasks of natural language processing. However, it suffers from many problems. In this paper, we address the problem of the unseen relationships between words in Arabic WordNet. More precisely, we focus on the ability for new relationships to be learned ‘automatically’ in Arabic WordNet from existing relationships. Using the Neural Tensor Network, we investigate how it can be an advantageous technique to fill the relationship gaps between Arabic WordNet words. With minimum resources, this model delivers meaningful results. The critical component is how to represent the entities of Arabic WordNet. For that, we use AraVec, a set of pre-trained distributed word representation for the Arabic language. We show how much it helps to use these vectors for initialization. We evaluated the model, using a number of tests which reveal that semantically-initialized vectors provide considerable greater accuracy than randomly initialized ones.

**Keywords.** Arabic WordNet, natural language processing, neural tensor network, AraVec, word representation, word embedding.

## 1 Introduction

Arabic WordNet (AWN) [2, 12, 5, 6] is a lexical database for the Arabic language. It has been

developed following the development process of Princeton WordNet (WN) [15, 24, 25] and Euro WordNet [36]. There is a degree of uncertainty around the term *wordnet*. It was created as a lexical database, however, some researchers refer to it as a semantic network because of its hierarchy in the way entities are seen as nodes and connected with edges [16]. Others define it as an ontology regarding some specific relations between conceptual categories [9]. However, these denotations cannot disguise the fact that many wordnets represent the largest publicly available lexical resource for many language. AWN is one of them. It is widely used in various fields of Natural Language Processing (NLP). Therefore, a considerable volume of research has been published to improve its content, in term of quantity [1, 3, 29] and quality [2, 9]. Along with this growth, however, there is increasing concern over the incompleteness and lack of the relationships in AWN.

These missing inter-relationships should be able to be addressed through the development of auto-reasoning within AWN. Auto-reasoning is most commonly seen in people, in what is usually called ‘commonsense reasoning’. For

instance, if a new species of plant is discovered, we do not have to specify that it has leaves. That is learned 'commonsense' knowledge which people can impute from existing knowledge without reference to external resources, otherwise defined by Davis *et al.* [10] as a simulation of how humans think and react in their every-day situations. It is therefore essential to look for missing relationships to improve AWN.

Much of the current research is focused on enhancements of existing knowledge bases (KB), using patterns or classifiers applied to external corpus, regardless of the type of KB [39, 21, 37]. This approach has limitations however, given that not all the embodied knowledge in a text is explicitly presented, and particularly, in respect of the Arabic language, as it is a low-resource language.

Our major objective is to find a way to predict new relations from existing ones in the AWN. To do this, we embrace the Neural Tensor Network (NTN) proposed by [32] because it gave better results when they applied it to WN and Freebase [7]. Also, It uses only the database for the prediction, no external resources. NTN operates by transforming entities into vectors – one vector representing the distribution of an entity in the database and its relationship with others. Relationships between entities are then expressed using a group of parameters within the NTN. This word-embedding feature makes the model more accurate when compared with other models.

We have structured this article in six sections, including this introductory section. Section 2 will describe the AWN and what version are we going to work on. Related works will be detailed in section 3. Section 4 will describe the word embedding technique that we use and the NTN model. An outline of some of the tests used and the evaluation of their results is provided in section 5, with our conclusions following in section 6.

## 2 Overview of Arabic WordNet

Arabic WordNet is a large lexical database of Arabic, which is available for public use, free of charge<sup>1</sup>. It comprises 5 parts of speech: nouns,

<sup>1</sup><http://compling.hss.ntu.edu.sg/omw/>

verbs, adverbs, adjectives, and adjective satellites, which are categorized into groups of synonyms called *synsets*. Each synset expresses a distinct sense or concept. They are interconnected by semantic and lexical relations. It is freely and publicly available for many tasks of NLP concerning the Arabic language. Currently, there are two versions of AWN. The version we are interested in is the LMF version. It is structured under the LMF standard, which makes it a practical tool for computational linguistics.

We find in the LMF Version<sup>2</sup> 60,154 entities in total. Most of them are verbs (42,298 entities) and nouns (16,432 entities). The rest are adverbs with 771, adjectives with 270, and adjective satellites with 386. With this variety, entities are connected with only 41,135 relations (5 type of relations). Our attention has been drawn to make progress in this version since it is rich in terms of entities but not in term of connection between them. First, we are going to cite some interesting works in this field.

## 3 Related Work

There is a large and growing body of literature which has been dedicated to an exposition of the incompleteness of existing KBs. The open-source KB, Freebase, before it closed, only held information regarding the nationality and birthplace of a small minority of those listed on it (25 and 29 percent, respectively) [11]. Studies of other KBs such as DBpedia [4] and YAGO [13] reveal that up to 99 percent of categorizes lack at least one property that others in the same class possess [34]. Galarraga *et al.* [17] discovered in 2016 that Wikidata knows the father of only 2 percent of all the people in it. For that, many attempts have been made to address this issue [14, 31]. Most of them used external resources in order to mine relevant rules such as if *X has children* then probably *X is married*. Thus, they can impute new facts and relations between entities. Use of external resources is an important factor but this can pose problems for AWN, given the low-resource nature of the Arabic language when compared to other like English.

<sup>2</sup>For the rest of the paper, AWN will refer to the LMF version.

There is nonetheless a significant minority of projects which have focused solely on improvements to a KB itself. In their major study, Nickel *et al.* [26] presented RESCAL, an approach based on the factorization of a three-way tensor. The main idea is to match the latent semantics of entities and relations, where a KB was regarded as a three-dimensional tensor. The matching score of a triplet is given by a bilinear function. Similarly, Jenatton *et al.* [20] proposed another tensor approach to model multi-relational datasets. It transformed entities into vectors and set a matrix for each relation. To reduce the number of parameters and avoid overfitting, relations were represented as a combination of latent factors. As Jenatton *et al.* observe, the bilinear structure of a latent factor model can reveal unseen shared similarities between different relation types.

The impact of tensor factorization has been explored in several studies of NLP. Setiawan *et al.* [30], He *et al.* [19], and Qiu *et al.* [28] proposed a different type of neural network that includes tensor decomposition. Sutskever *et al.* [35] proposed the Bayesian Clustered Tensor Factorization (BCFT). Their principal objective was to find interpretable structures within a KB and predict the accuracy of unobserved relationships within that KB. This model clusters entities and relations to share statistical strength through a three-way interaction using a bayesian non-parametric method. Socher *et al.* [32] found a way to predict new relations based on others. They developed the Neural Tensor Network (NTN), which is a tensor decomposition in a neural network architecture. It has been applied to WN and Freebase. Each relation in the two resources has its own tensor parameters and each entity is represented as an average of its constituent word vectors in high dimension.

Socher *et al.* also claim that representing entities with a single vector does not allow the sharing of statistical strength between similar entities. For instance, 'living room' and 'dining room' are two entities that have much in common, yet previous approaches are likely to embed each one separately. So, they embedded each of 'living', 'room', and 'dining' and then built the representations for entities as the average of the

vectors of the three words. For that purpose, they initialized word vectors with pre-trained vectors using a word embedding model, demonstrating that this can increase the reasoning accuracy. In the same vein, Bordes *et al.* [8] proposed the Structured Embedding model. The main idea is that two entities of a correct triple should be closely related to each other in the relation space. Hence, two entities are defined by two (-head, and tail) relation-specific matrices and the score function correlates with the degree of relationship between the words.

While there is are many different models for word embedding, all of which work in their different ways, ultimately, in every model words will be represented as vectors in a continuous space. In 2013, Mikolov *et al.* [23] developed the famous word2vec, a model for word embedding with two different architectures, the continuous bag of words (CBOW), and the skip-gram (SKIP-G). The difference between them is that the first one learns to predict the word from its context and the second one learns to predict the context from the word. But, either way, they both represent the word in vector space according to its local context. CBOW is faster to train and works better with frequent words. SKIP-G performs well with a small training set and works with rare words.

A year later, Pennington *et al.* [27] presented an alternative model called GloVe. The major difference between GloVe and word2vec is that it leans by constructing a co-occurrence matrix: in other words how frequently a word appears in a context. In several cases, word2vec has proven to be a more successful algorithm for unsupervised learning of semantic similarity and relatedness, when compared with GloVe [22].

Recently, in 2016, a group of researchers on Facebook proposed fastText [18]. Rather than a word by word approach, like word2vec and GloVe, it breaks words into n-grams. For instance, the tri-grams for the word *fruit* is *fru*, *rui*, and *uit* and the vector of the word will be the sum of the n-grams. This approach provides a significant advantage in respect of rare words, and words that are not presented in the training set. The authors claim that this outperforms word2vec since it can present a vector for any word, something that neither

word2vec nor GloVe can do. However, research has consistently shown that both these models require large datasets for the training process.

In a major study concerning the Arabic language, Zahran *et al.* [38] built a vectorized word embedding for the Modern Standard Arabic (MSA) using CBOW, SKIP-G, and GloVe. They also evaluated them: intrinsically, using the task of standard word similarity; and extrinsically, using two NLP application, Information Retrieval, and Short Answer Grading.

Soliman *et al.* [33] presented the AraVec, a set of pre-trained distributed word embedding for the Arabic language. Its first version has six word embedding models trained on three datasets (Wikipedia, Twitter, and texts from Arabic web pages) with more than 3,300M tokens. The second version now contains 12 models. The main idea is to present each one of the datasets with the two models of word2vec (CBOW and SKIP-G). This outperformed fastText because it has been trained on a large dataset includes tweets (Arabic dialects).

The most obvious finding to emerge from these studies is that the tensor approach is very effective in predicting accurately missing information in linguistics resources. Also, this has enhanced our understanding of representing entities as feature vectors to preserve the knowledge of the original data and present the interesting ability of generalizing new reasonable relations.

## 4 Proposed Model

In this section, we focus on how to complete missing relationships between words in AWN. As shown previously, the neural tensor network (NTN), alongside word embedding techniques, arise naturally with the representation of multi-relational data, like AWN. For that, first, we prepare all the entities in AWN and then build the model by training it and adjusting its parameters.

### 4.1 Word Embedding

Word embedding, as a process, is learned from data. There are two ways to obtain the word embedding. The first method takes random vectors and trains the model until the training loss function can decrease no further and the second takes pre-trained vectors. The high volume of data input required of the first model meant that the second one is better suited to our research, given the low-resource nature of Arabic.

Many previous cited works have similarly relied on the advantage of using pre-trained word embedding vectors in their works, due to many factors. The rationale behind this is it is either because of the insufficient data available to learn truly powerful features (less-resourced language like the Arabic) or the disposition of powerful materials required for the calculation (training word2vec with a normal laptop could take hours). Besides, using a pre-trained word embedding vectors shorten the training time and make a better quality of word vectors. Another cause specific to our goal is that the strength shared between the words in a corpus is stronger than the strength shared between them in AWN.

There are several of pre-trained word embedding datasets with different models at the disposal of the NLP community. Our research used AraVec developed by Soliman *et al.* [33]. It is a set of word embedding models with 100 and 300 dimensions. It has been trained on different datasets. We combined the model CBOW that has been trained on Wikipedia and Web. Entities in AWN are either single or multi-word expression. AraVec made this task easiest because the vector of an expression is represented by the average of its word vectors. Once we indexed each word in AWN with its appropriate vector, we use them to train the NTN.

### 4.2 Neural Tensor Network

We adopted the NTN model provided by Socher *et al.* [32]. Not only has NTN performed well when used with NW and Freebase, it also outperforms other neural networks, such as the single and bilinear networks in tests, due to its ability to link entities across multiple dimensions.

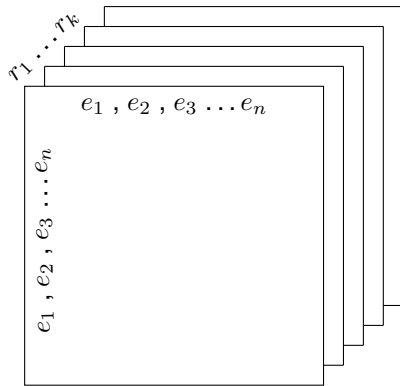


Fig. 1. Illustration of the tensor model

As we can see, figure 1 represent entities  $e_n$  and relations  $r_k$  as triples. This is called a three-way tensor. We can tell that a *slice* of the tensor is defined by a matrix when  $r$  is fixed. If we have two entities  $e_1$  and  $e_2$  with a relation  $r$ , the score of their relationship is a function  $s(e_1, r, e_2)$ :

$$s(e_1, r, e_2) = u_r^T f(e_1^T W_r^{[1:k]} e_2 + V_r, \begin{bmatrix} e_1 \\ e_2 \end{bmatrix} + b_r), \quad (1)$$

$F$  is a  $\tanh$  nonlinear function.  $e_1^T W_r^{[1:k]} e_2$  is a tensor product where  $W^{[1:k]}$  is the tensor and  $k$  represent the slices of the tensor.  $U_r$ ,  $V_r$ , and  $b_r$  are the parameters of a standard neural network. The non-linearity of this model is the main advantage because it joints the entities directly through multiplicity. Each relation  $r$  has its own parameters as it is illustrated in the figure 1. A visualization of the tensor layer is shown in figure 2 provided in [32].

The model needs to be trained. This yield to adjust all the parameters  $U$ ,  $E$ ,  $W$ ,  $V$ , and  $b$  by minimizing the hinge-loss function 2. The main idea is to give a higher score to a positive triple  $(e_1^{(i)}, r^{(i)}, e_2^{(i)})$  than a negative triple  $(e_1^{(i)}, r^{(i)}, e_c)$ , in which  $e_c$  is a random entity:

$$J(\Omega) = \sum_{i=1}^N \sum_{c=1}^C \max(0, 1 - s(T^{(i)}) + s(T_c^{(i)})) + \lambda \|\Omega\|_2^2. \quad (2)$$

With  $\Omega$  represents all the parameters of the network. 1 is the margin and  $s$  is the score.

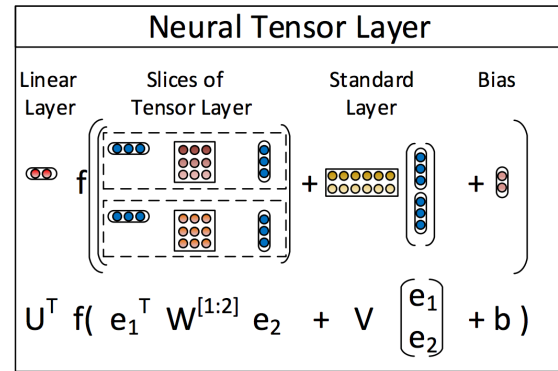


Fig. 2. Neural tensor Layer parameters with 2 slices

$N$  and  $C$  are the set of positive and negative samples, respectively.  $T^{(i)}$  and  $T_c^{(i)}$  are the positive (correct) and negative triple, respectively.  $\lambda$  is the regularization parameter to prevent overfitting. To update these parameters and calculate  $\partial J / \partial \Omega$ , we use the backpropagation algorithm.

## 5 Test and Evaluation

The concept is easier to understand using an example. If <sup>3</sup> (Bosporus) is a (strait) and a is a (canal) then is a too. This is simply how the model will predict new relations in a transitive way. To make the semantic and lexical sharing stronger, entities should be presented with as many relations as possible. For that, we keep only the accurate ones that are presented with more than 2 relations. In total, we have 41,122 connected triples with 5 type of relations. We did not work with the *antonym* relation, and we concatenated the *hyponym/hypernym* (is a) and *HasInstance/isInstance* (instance) as one relation, respectively. We end up with 3 types of relations and we split the data as showed in table 1.

We do not have a large data to be able to train the model perfectly, so we used the k-fold cross validation (we choose  $k = 10$ ) to better entertain the training and the test sets. For each  $k$ , we combine the training and validation sets and run

<sup>3</sup>Arabic words are followed by their transliteration using the transliteration system of  $\mathbb{A}\mathbb{T}\mathbb{E}\mathbb{X}$  and their English translations in brackets.

**Table 1.** Statistics of the AWN preparation

Database	Relations	Training set	Validation set	Test set
Arabic	3	28,272	2,570	10,280
WordNet				

the algorithm once again. Also, we notice that many words do not have any relations at all. So, we created manual associations with at least one relation between a word and a triple and had that verified by a lexicographer. To add more precision to the model and to force it to focus on harder cases, we add some of them to the validation and the rest to the test set, since the main goal is to predict new relations. Negative examples are created randomly by switching one entity in a correct triple in the training set.

Another matter that needs attention is the complexity of the model. The complexity is represented by the number of the parameters within the tensor that need to be trained. It lies exactly in the tensor. To reduce complexity in the model, we choose to work with only 3 slices of tensor. For instance, if the dimension of the entities vectors is  $d = 100$  ( $e_i \in \mathbb{R}^{100}$ ) then the tensor  $W_R^{[1:k]} \in \mathbb{R}^{d*d*k}$  ( $10000 * k$ ) and the standard parameter  $V_R \in \mathbb{R}^{2*d}$  ( $2d * k$ ). In the end, with 3 slices, we have over 30,000 parameters to train. For that, we could not add a new slice otherwise we have to add new relations between all the entities.

In general, the accuracy of the model is based on how correct new triples are.

**Table 2.** Recall and precision of different relations in AWN

Relations	Recall (%)	Precision (%)
Is a	29.3	71.8
Instance	32.7	45.7
Similar	13.7	32.5

The disparity as shown in table 2 lies in the difference between the relations in terms of the number of triples in the training set. There is also the complexity of the relation. For instance, it is easier to reason for example over the *is a*

relation than the *instance*. Furthermore, the vector representation takes part of this disparity.

Table 3 show how much the initialization with pre-trained vectors help.

**Table 3.** Variance between different entities representations

Initialization	Accuracy (%)
Randomly	52
Pre-trained with 100 dimensions	<b>71</b>
Pre-trained with 300 dimensions	68

We tested the model with a random initialization and the two sets of AraVec: 100 and 300 dimensions. Both of the distribution of AraVec gave almost the same results but the random initialization showed lower accuracy. This due to the multi-word expressions presented in the AWN. We considered by presenting the multi-word expressions as the average of the words that combine them it will strengthen the semantic sharing between them. We can explain the low accuracy by the lack of the triples in the training set. Most importantly, this model can reason without recourse to external resources.

## 6 Conclusion

Our research addressed the incompleteness of the Arabic WordNet using the natural tensor network as a tool. Results achieved using the NTN have improved through the incorporation of pre-trained word vectors. The intersection between relations made via a tensor layer. The study has demonstrated meaningful results extending the number of relations within Arabic WordNet. Such an automated process inevitably brings with it the risk of adding wrong information. Further research will be focused on error detection in the Arabic WordNet and an investigation of the quality of its information.

## References

1. Abouenour, L., Bouzoubaa, K., & Rosso, P. (2010). Using the yago ontology as a resource for the enrichment of named entities in arabic

- wordnet. *Proceedings of The Seventh International Conference on Language Resources and Evaluation (LREC 2010) Workshop on Language Resources and Human Language Technology for Semitic Languages*, pp. 27–31.
2. **Abouenour, L., Bouzoubaa, K., & Rosso, P. (2013).** On the evaluation and improvement of arabic wordnet coverage and usability. *Language resources and evaluation*, pp. 891–917.
  3. **Alkhalifa, M. & Rodríguez, H. (2010).** Automatically extending named entities coverage of arabic wordnet using wikipedia. *International Journal on Information and Communication Technologies*, pp. 20–36.
  4. **Auer, S., Bizer, C., Kobilarov, G., Lehmann, J., Cyganiak, R., & Ives, Z. G. (2007).** Dbpedia: A nucleus for a web of open data. *The Semantic Web, 6th International Semantic Web Conference, 2nd Asian Semantic Web Conference, ISWC 2007 + ASWC 2007, Busan, Korea, November 11-15, 2007.*, pp. 722–735.
  5. **Black, W., Elkateb, S., Rodríguez, H., Alkhalifa, M., Vossen, P., Pease, A., & Fellbaum, C. (2006).** Introducing the arabic wordnet project. *Proceedings of the third international WordNet conference*, pp. 295–300.
  6. **Black, W., Sabri, E., Horacio, R., Musa, A., Piek, V., Adam, P., & Christiane, F. (2006).** Introducing the arabic wordnet project. *In Proceedings of the third international WordNet conference*.
  7. **Bollacker, K., Evans, C., Paritosh, P., Sturge, T., & Taylor, J. (2008).** Freebase: A collaboratively created graph database for structuring human knowledge. *Proceedings of the 2008 ACM SIGMOD International Conference on Management of Data*, pp. 1247–1250.
  8. **Bordes, A., Weston, J., Collobert, R., Bengio, Y., et al. (2011).** Learning structured embeddings of knowledge bases. *AAAI*, pp. 6.
  9. **Boudabous, M. M., Kammoun, N. C., Khedher, N., Belguith, L. H., & Sadat, F. (2013).** Arabic wordnet semantic relations enrichment through morpho-lexical patterns. *Communications, Signal Processing, and their Applications (ICCSPA), 2013 1st International Conference on*, pp. 1–6.
  10. **Davis, E. & Marcus, G. (2015).** Commonsense reasoning and commonsense knowledge in artificial intelligence. *Communications of the ACM*, pp. 92–103.
  11. **Dong, X., Gabrilovich, E., Heitz, G., Horn, W., Lao, N., Murphy, K., Strohmman, T., Sun, S., & Zhang, W. (2014).** Knowledge vault: A web-scale approach to probabilistic knowledge fusion. *Proceedings of the 20th ACM SIGKDD international conference on Knowledge discovery and data mining*, pp. 601–610.
  12. **Elkateb, S., Black, W., Rodríguez, H., Alkhalifa, M., Vossen, P., Pease, A., & Fellbaum, C. (2006).** Building a wordnet for arabic. *Proceedings of The fifth international conference on Language Resources and Evaluation (LREC 2006)*, pp. 22–28.
  13. **Fabian, M., Gjergji, K., Gerhard, W., et al. (2007).** Yago: A core of semantic knowledge unifying wordnet and wikipedia. *16th International World Wide Web Conference, WWW*, pp. 697–706.
  14. **Fader, A., Soderland, S., & Etzioni, O. (2011).** Identifying relations for open information extraction. *Proceedings of the 2011 Conference on Empirical Methods in Natural Language Processing, EMNLP 2011, 27-31 July 2011, John McIntyre Conference Centre, Edinburgh, UK, A meeting of SIGDAT, a Special Interest Group of the ACL*, pp. 1535–1545.
  15. **Fellbaum, C. (1998).** *WordNet*. Wiley Online Library.
  16. **Fontenelle, T. (2012).** Wordnet, framenet and other semantic networks in the international journal of lexicography – the net result? *International Journal of Lexicography*, pp. 437–449.
  17. **Galárraga, L., Razniewski, S., Amarilli, A., & Suchanek, F. M. (2017).** Predicting completeness in knowledge bases. *Proceedings of the Tenth ACM International Conference on Web Search and Data Mining*, pp. 375–383.
  18. **Grave, E., Mikolov, T., Joulin, A., & Bojanowski, P. (2017).** Bag of tricks for efficient text classification. *Proceedings of the 15th Conference of the European Chapter of the Association for Computational Linguistics*, pp. 427–431.
  19. **He, X., Xu, H., Sun, X., Deng, J., & Li, J. (2017).** Abircnn with neural tensor network for answer selection. *Proc. Int. Joint Conf. Neural Networks (IJCNN)*, pp. 2582–2589.
  20. **Jenatton, R., Roux, N. L., Bordes, A., & Obozinski, G. R. (2012).** A latent factor model

for highly multi-relational data. *Advances in Neural Information Processing Systems*, pp. 3167–3175.

21. **Kaushik, N. & Chatterjee, N. (2016).** A practical approach for term and relationship extraction for automatic ontology creation from agricultural text. *Proc. Int. Conf. Information Technology (ICIT)*, pp. 241–247.
22. **Major, V., Surkis, A., & Aphinyanaphongs, Y. (2017).** Utility of general and specific word embeddings for classifying translational stages of research. *arXiv preprint arXiv:1705.06262*.
23. **Mikolov, T., Chen, K., Corrado, G., & Dean, J. (2013).** Efficient estimation of word representations in vector space. *arXiv preprint arXiv:1301.3781*.
24. **Miller, G. A. (1995).** Wordnet: a lexical database for english. *Communications of the ACM*, pp. 39–41.
25. **Miller, G. A., Beckwith, R., Fellbaum, C., Gross, D., & Miller, K. J. (1990).** Introduction to wordnet: An on-line lexical database. *International journal of lexicography*, pp. 235–244.
26. **Nickel, M., Tresp, V., & Kriegel, H.-P. (2011).** A three-way model for collective learning on multi-relational data. *ICML*, pp. 809–816.
27. **Pennington, J., Socher, R., & Manning, C. (2014).** Glove: Global vectors for word representation. *Proceedings of the 2014 conference on empirical methods in natural language processing (EMNLP)*, pp. 1532–1543.
28. **Qiu, X. & Huang, X. (2015).** Convolutional neural tensor network architecture for community-based question answering. *Proceedings of the Twenty-Fourth International Joint Conference on Artificial Intelligence, IJCAI 2015, Buenos Aires, Argentina, July 25-31, 2015*, pp. 1305–1311.
29. **Rodríguez, H., Farwell, D., Ferreres, J., Bertran, M., Alkhalifa, M., & Martí, M. A. (2008).** Arabic wordnet: Semi-automatic extensions using bayesian inference. *LREC*.
30. **Setiawan, H., Huang, Z., Devlin, J., Lamar, T., Zbib, R., Schwartz, R., & Makhoul, J. (2015).** Statistical machine translation features with multitask tensor networks. *arXiv preprint arXiv:1506.00698*.
31. **Snow, R., Jurafsky, D., & Ng, A. Y. (2004).** Learning syntactic patterns for automatic hypernym discovery. *Advances in Neural Information Processing Systems 17 [Neural Information Processing Systems, NIPS 2004, December 13-18, 2004, Vancouver, British Columbia, Canada]*, pp. 1297–1304.
32. **Socher, R., Chen, D., Manning, C. D., & Ng, A. Y. (2013).** Reasoning with neural tensor networks for knowledge base completion. *Advances in Neural Information Processing Systems 26: 27th Annual Conference on Neural Information Processing Systems 2013. Proceedings of a meeting held December 5-8, 2013, Lake Tahoe, Nevada, United States.*, pp. 926–934.
33. **Soliman, A. B., Eissa, K., & El-Beltagy, S. R. (2017).** Aravec: A set of arabic word embedding models for use in arabic nlp. *Procedia Computer Science*, pp. 256–265.
34. **Suchanek, F. M., Gross-Amblard, D., & Abiteboul, S. (2011).** Watermarking for ontologies. *International Semantic Web Conference*, pp. 697–713.
35. **Sutskever, I., Salakhutdinov, R., & Tenenbaum, J. B. (2009).** Modelling relational data using bayesian clustered tensor factorization. *Advances in Neural Information Processing Systems 22: 23rd Annual Conference on Neural Information Processing Systems 2009. Proceedings of a meeting held 7-10 December 2009, Vancouver, British Columbia, Canada.*, pp. 1821–1828.
36. **Vossen, P. (1998).** *EuroWordNet: A multilingual database with lexical semantic networks*. Kluwer Academic Publishers.
37. **Wang, M., Li, L., & Huang, F. (2014).** Semi-supervised chinese open entity relation extraction. *Proc. IEEE 3rd Int. Conf. Cloud Computing and Intelligence Systems*, pp. 415–420.
38. **Zahran, M. A., Magooda, A., Mahgoub, A. Y., Raafat, H., Rashwan, M., & Atyia, A. (2015).** Word representations in vector space and their applications for arabic. *International Conference on Intelligent Text Processing and Computational Linguistics*, pp. 430–443.
39. **Zhuang, T., Wang, P., Zhang, Y., & Zhang, Z. (2017).** A novel method for open relation extraction from public announcements of chinese listed companies. *Proc. Fifth Int. Conf. Advanced Cloud and Big Data (CBD)*, pp. 200–205.

Article received on 17/01/2019; accepted on 04/03/2019.  
Corresponding author is Mohamed Ali Batita.



# Refining Concepts by Machine Learning

Marek Menšík<sup>1</sup>, Marie Duží<sup>1</sup>, Adam Albert<sup>1</sup>, Vojtěch Patschka<sup>1</sup>, Miroslav Pajr<sup>2</sup>

<sup>1</sup> VSB-Technical University of Ostrava,  
Department of Computer Science,  
Czech Republic

<sup>2</sup> Silesian University in Opava,  
Institute of Computer Science,  
Czech Republic

{marek.mensik, marie.duzi}@vsb.cz

**Abstract.** In this paper we deal with machine learning methods and algorithms applied in learning simple concepts by their refining or explication. The method of refining a simple concept of an object  $O$  consists in discovering a molecular concept that defines the same or a very similar object to the object  $O$ . Typically, such a molecular concept is a professional definition of the object, for instance a biological definition according to taxonomy, or legal definition of roles, acts, etc. Our background theory is Transparent Intensional Logic (TIL). In TIL concepts are explicated as abstract procedures encoded by natural language terms. These procedures are defined as six kinds of TIL constructions. First, we briefly introduce the method of learning with a supervisor that is applied in our case. Then we describe the algorithm 'Framework' together with heuristic methods applied by it. The heuristics is based on a plausible supply of positive and negative (near-miss) examples by which learner's hypotheses are refined and adjusted. Given a positive example, the learner refines the hypothesis learnt so far, while a near-miss example triggers specialization. Our heuristic methods deal with the way refinement is applied, which includes also its special cases generalization and specialization.

**Keywords.** Machine learning, supervisor, transparent intensional logic, TIL, refinement, generalization, specialization, hypothesis, heuristics.

## 1 Introduction

The method of supervised machine learning enables the agents in a multi-agent system to adjust their ontology and increase their knowledge. In [13] the method has been applied to learning the

concept of a property that classifies geometric figures such as lancet arches.

In this paper we deal with natural language processing, which is an interdisciplinary discipline involving linguistics, logic and computer science. The goal of this paper is to describe the application of machine learning methods in agents' learning simple concepts by their refinement or explication. *Refinement* is rigorously defined below. Briefly, by refining a simple concept of an object  $O$ , we mean discovering a molecular concept that defines the same object  $O$ . In mathematics we use definitions like "a *group* is a set  $G$  equipped with a binary operation that combines any two elements of  $G$  to form another element of  $G$  in such a way that group axioms are satisfied, namely associativity, existence of the neutral element in  $G$  and invertibility."

Here the simple concept to be refined is that of a 'group'. The molecular concept refining 'group' is encoded by the right side of the definition, namely 'a set  $G$  equipped with a binary operation that combines any two elements of  $G$  to form another element of  $G$  in such a way that group axioms are satisfied, namely associativity, existence of the neutral element in  $G$  and invertibility'. In case of *empirical* concepts, it is more plausible to speak about *explication*. The reason is this.

To say that a molecular concept  $C$  is a refinement of a simple empirical concept  $D$  is risky. It would be a refinement only if the molecular concept  $C$  was *equivalent* to the original concept

$D$ , which means that both are the concepts of the same object  $O$ .

However, in the most interesting cases of empirical concepts of PWS-*intensions* we use a Carnapian *explication* rather than a definition proper, and then equivalence is surely not guaranteed.<sup>1</sup> Rather, a new molecular concept  $C$  (explicatum) should define an object  $O$  that is as close as possible to the object referred to by an inexact prescientific concept  $D$  (explicandum). In *Meaning and Necessity* (1947), Carnap characterizes explication as follows:

The task of making more exact a vague or not quite exact concept used in everyday life or in an earlier stage of scientific or logical development, or rather of replacing it by a newly constructed, more exact concept, belongs among the most important tasks of logical analysis and logical construction. We call this the task of explicating, or of giving an *explication* for, the earlier concept [1], (pp. 7-8) Keeping this difference in mind, in this paper we use the term ‘refinement’ for both cases, including explication of empirical concepts, because in our sample example of explicating the concept of myopia this simplification is harmless.

Our background theory is Transparent Intensional Logic (TIL) with its *procedural* (as opposed to set-theoretical) semantics. In TIL we explicate concepts procedurally. They are abstract structured *procedures* assigned to natural language terms as their meanings. In this way *structured meanings* are formalized in a fine-grained way as so-called TIL *constructions* so that almost all the semantically salient features can be successfully dealt with.

To this end we use the so-called Normal Translation Algorithm (NTA) that processes text data and produces TIL constructions as their

meanings.<sup>2</sup> Having a meaning procedure, we can apply logic to prove what is entailed by it, compute the object (if any) produced by the procedure, deal with its structure, etc.

However, there is a problem of understanding simple or atomic concepts that are expressed by semantically simple terms like ‘cat’, ‘dog’, ‘myopia’, etc. They are basic ‘building blocks’ of molecular concepts, and as such they are formalized just by the simplest procedure Trivialization of a given object  $O$ , ‘ $^0O$ ’ in symbols, that refers to the object  $O$  and makes it available to other molecular procedures to operate on it. In proof-theoretic semantics the meaning of atomic terms is given by the rules that determine how to use them in proofs.<sup>3</sup> This works well in the language of mathematics and logic.

However, in natural language the ‘meaning as proof’ semantics is much less successful. For these reasons we decided to apply supervised machine learning methods.

The issue is this. When processing a natural language text, our agents learn structured TIL procedures encoded by sentences. For instance, the sentence “Tom has myopia” translates into the TIL procedure  $\lambda w \lambda t [^0Myopia_{wt} \ ^0Tom]$ .

It can be viewed as an instruction how in any possible world ( $\lambda w$ ) and time ( $\lambda t$ ) evaluate the truth-conditions of the sentence, which consists of these steps:

- Take the individual Tom:  $^0Tom$
- Take the property of having Myopia:  $^0Myopia$
- Extensionalize the property with respect to world  $w$  and time  $t$  of evaluation:  $^0Myopia_{wt}$
- Produce a truth-value by checking whether Tom has this property at the world  $w$  and time  $t$  of evaluation:  $[^0Myopia_{wt} \ ^0Tom]$

<sup>1</sup> As an example, consider the simple concept of a *planet*. Which property falls under this concept? For sure, it is a property of individuals such that being a celestial body is a requisite of the property. Necessarily, if any individual  $x$  happens to be a planet, then  $x$  is a celestial body. However, which are the other requisites? One of the results of IAU 2006 General Assembly in Prague was the resolution 5A on ‘Definition of Planet’. A ‘planet’ was defined as a celestial body that (a) is in orbit around the Sun, (b) has sufficient mass for its self-gravity to overcome rigid body forces so that it assumes a hydrostatic equilibrium (nearly round) shape,

and (c) has cleared the neighbourhood around its orbit. Is this definition a refinement of the original concept of a planet? No, it is just an explication. As a secondary result, it also decided that the modifier ‘dwarf’ is privative with respect to ‘planet’ (see [5]); a dwarf planet like Pluto is not a planet. For details see, IAU International Astronomical Union 0603 Press Release: <https://www.iau.org/news/pressreleases/detail/iau0603/>

<sup>2</sup> For details, see [10], [12].

<sup>3</sup> See, for instance, [9].

So far so good. We can derive that somebody has myopia, but this piece of information does not suffice to derive, for instance, that Tom has problems with impaired vision, needs negative dioptre correction, etc.

We need to refine the simple concept <sup>0</sup>*Myopia* to learn in more details what 'myopia' is. In other words, we want to define the property of having myopia. To this end we try to extract from natural language texts the collection of so-called *requisites* that together define the property. Hence, the supervisor looks for sentences like "Myopia (also called near-sightedness) is the most common cause of impaired vision in people under age 40". Based on this piece of information the agent makes a *hypothesis* that among the requisites of myopia there are 'near-sightedness' and 'impaired vision'.

This is a positive example. Furthermore, we can read sentences like "Myopia is not caused by nerve trauma; rather, it occurs when the eyeball is too long, relative to the focusing power of the cornea and lens of the eye. This causes light rays to focus at a point in front of the retina, rather than directly on its surface. Near-sightedness also can be caused by the cornea and/or lens being too curved for the length of the eyeball. In some cases, myopia is due to a combination of these factors." The supervisor should extract a negative example that myopia is not caused by nerve trauma and a collection of positive examples like 'too long eyeball', 'wrong focusing', etc.

The algorithm of the learning process is based on such positive and negative examples. Given a positive example, the hypotheses are adjusted so that concepts of other requisites or typical properties are inserted. Negative (also 'near-miss') examples serve to the adjustment of the hypothesis (learnt so far) by *specialization* that excludes non-plausible elements. As a special case of refinement, we can also apply *generalization*. This is the case of inserting a more general concept in addition to some special constituents of the hypothesis. For instance, the degree of myopia is described in terms of the power of the ideal correction, which is measured in dioptres.

Now the agent can extract information like this. "Low myopia usually describes myopia of -3.00 dioptres or less (i.e. closer to 0.00), moderate myopia is between -3.00 and -6.00 dioptres, and

high myopia is the degree -6.00 or more." By generalization we obtain information that myopia is corrected by negative dioptres. The rest of the paper is organized as follows.

In Section 2 we summarize foundations of TIL to describe logical machinery that we need in the rest of the paper. Section 3 introduces the principles of supervised machine learning. In Section 4 we deal with heuristic methods that are used to adjust and enrich agents' knowledge base. In Section 5, an example of using the algorithm of machine learning together with TIL formalization is adduced. Finally, concluding remarks can be found in Section 6.

## 2 Foundations of Transparent Intensional Logic (TIL)

Since the TIL logical system has been introduced in numerous papers and two books, see, for instance [2, 3, 4, 5, 7, 8, 17], here we just briefly summarise the main principles of a TIL fragment that we need for the purposes of this paper.

TIL is a partial, typed hyperintensional lambda calculus with *procedural* as opposed to set-theoretical denotational semantics. The terms of the TIL language denote abstract procedures that produce set-theoretical mappings (functions-in-extension) or lower-order procedures. These procedures are rigorously defined as TIL *constructions*. Being procedural objects, constructions can be executed in order to operate on input objects (of a lower-order type) and produce the object (if any) they are typed to produce, while non-procedural objects, i.e. non-constructions, cannot be executed. There are two atomic constructions that present input objects to be operated on.

They are *Trivialization* and *Variables*. The operational sense of Trivialization is similar to that of constants in formal languages. Trivialization presents an object *X* without the mediation of any other procedures. Using the terminology of programming languages, the Trivialization of *X*, <sup>0</sup>*X* in symbols, is just a pointer that *refers* to *X*. Variables produce objects dependently on valuations; they *v-construct*. We adopt an objectual variant of the Tarskian conception of variables. To each type countably many variables are assigned

that range over this particular type. Objects of each type can be arranged into infinitely many sequences.

The valuation  $v$  selects one such sequence of objects of the respective type, and the first variable  $v$ -constructs the first object of the sequence, the second variable  $v$ -constructs the second object of the sequence, and so on. Thus, the execution of a Trivialization or a variable never fails to produce an object. However, the execution of some of the molecular constructions can fail to present an object of the type they are typed to produce. When this happens, we say that such constructions are  $v$ -improper.

There are two dual molecular constructions, which correspond to  $\lambda$ -abstraction and application in  $\lambda$ -calculi, namely *Closure* and *Composition*.  $(\lambda\text{-})$ Closure,  $[\lambda x_1 \dots x_n X]$ , transforms into the very procedure of producing a function by abstracting over the values of the variables  $x_1, \dots, x_n$ . The Closure  $[\lambda x_1 \dots x_m Y]$  is not  $v$ -improper for any valuation  $v$ , as it always  $v$ -constructs a function. Composition,  $[X X_1 \dots X_n]$ , is the very procedure of applying a function produced by the procedure  $X$  to the tuple-argument (if any) produced by the procedures  $X_1, \dots, X_n$ . While Closure never fails to produce a function, Composition is  $v$ -improper if one or more of its constituents  $X, X_1, \dots, X_n$  are  $v$ -improper.

This happens when a partial function  $f$  is applied to an argument  $a$  such that the function  $f$  is not defined at  $a$ . Another cause of improperness can be type-theoretical incoherence of the Composition. For instance, the proposition that the number 5 is a student does not have a truth-value at any world  $w$  and time  $t$  of evaluation, because the property of being a student is the property of individuals rather than numbers. Hence the application of the (extensionalized) property of being a student to the number 5 in a particular world  $w$  and time  $t$  of evaluation, in symbols  $[[[{}^0\text{Student } w]t]{}^05]$ , or  $[{}^0\text{Student}_{wt}{}^05]$  for short, is  $v$ -improper for every valuation  $v$  of the variables  $w$  (ranging over possible worlds) and  $t$  (ranging over times).

In what follows we define six kinds of TIL constructions, and afterwards the ramified hierarchy of types into which objects of TIL ontology are organised.

### Definition (Constructions)

- (i) Variables  $x, y$ , are constructions that construct objects (elements of their respective ranges) dependently on a valuation  $v$ ; they  $v$ -construct.
- (ii) Where  $X$  is an object whatsoever (even a construction),  ${}^0X$  is the construction Trivialization that constructs  $X$  without any change of  $X$ .
- (iii) Let  $X, Y_1, \dots, Y_n$  be arbitrary constructions. Then Composition  $[X Y_1 \dots Y_n]$  is the following construction. For any  $v$ , the Composition  $[X Y_1 \dots Y_n]$  is  $v$ -improper if at least one of the constructions  $X, Y_1, \dots, Y_n$  is  $v$ -improper by failing to  $v$ -construct anything, or if  $X$  does not  $v$ -construct a function that is defined at the  $n$ -tuple of objects  $v$ -constructed by  $Y_1, \dots, Y_n$ . If  $X$  does  $v$ -construct such a function, then  $[X Y_1 \dots Y_n]$   $v$ -constructs the value of this function at the  $n$ -tuple.
- (iv)  $(\lambda\text{-})$  Closure  $[\lambda x_1 \dots x_m Y]$  is the following construction. Let  $x_1, x_2, \dots, x_m$  be pair-wise distinct variables and  $Y$  a construction. Then  $[\lambda x_1 \dots x_m Y]$   $v$ -constructs the function  $f$  that takes any members  $B_1, \dots, B_m$  of the respective ranges of the variables  $x_1, \dots, x_m$  into the object (if any) that is  $v(B_1/x_1, \dots, B_m/x_m)$ -constructed by  $Y$ , where  $v(B_1/x_1, \dots, B_m/x_m)$  is like  $v$  except for assigning  $B_1$  to  $x_1, \dots, B_m$  to  $x_m$ .
- (v) Where  $X$  is an object whatsoever,  ${}^1X$  is the construction Single Execution that  $v$ -constructs what  $X$   $v$ -constructs. Thus, if  $X$  is a  $v$ -improper construction or not a construction as all,  ${}^1X$  is  $v$ -improper.
- (vi) Where  $X$  is an object whatsoever,  ${}^2X$  is the construction Double Execution. If  $X$  is not itself a construction, or if  $X$  does not  $v$ -construct a construction, or if  $X$   $v$ -constructs a  $v$ -improper construction, then  ${}^2X$  is  $v$ -improper. Otherwise  ${}^2X$   $v$ -constructs what is  $v$ -constructed by the construction  $v$ -constructed by  $X$ .
- (vii) Nothing is a construction, unless it so follows from (i) through (vi).

With constructions of constructions, constructions of functions, functions, and functional values in our stratified ontology, we need

to keep track of the traffic between multiple logical strata. The *ramified type hierarchy* does just that. The type of first-order objects includes all objects that are not constructions. Therefore, it includes not only the standard objects of individuals, truth-values, sets, etc., but also functions defined on possible worlds (i.e., the intensions germane to possible-world semantics). The type of second-order objects includes constructions of first-order objects and functions that have such constructions in their domain or range. The type of third-order objects includes constructions of first- and second-order objects and functions that have such constructions in their domain or range. And so on, ad infinitum.

**Definition (types of order  $n$ ).** Let  $B$  be a *base*, where a base is a collection of pair-wise disjoint, non-empty sets. Then:

**$T_1$  (types of order 1).**

- i) Every member of  $B$  is an elementary type of order 1 over  $B$ .
- ii) Let  $\alpha, \beta_1, \dots, \beta_m$  ( $m > 0$ ) be types of order 1 over  $B$ . Then the collection  $(\alpha \beta_1 \dots \beta_m)$  of all  $m$ -ary partial mappings from  $\beta_1 \times \dots \times \beta_m$  into  $\alpha$  is a functional type of order 1 over  $B$ .
- iii) Nothing is a type of order 1 over  $B$  unless it so follows from (i) and (ii).

**$C_n$  (constructions of order  $n$ )**

- i) Let  $x$  be a variable ranging over a type of order  $n$ . Then  $x$  is a construction of order  $n$  over  $B$ .
- ii) Let  $X$  be a member of a type of order  $n$ . Then  ${}^0X, {}^1X, {}^2X$  are constructions of order  $n$  over  $B$ .
- iii) Let  $X, X_1, \dots, X_m$  ( $m > 0$ ) be constructions of order  $n$  over  $B$ . Then  $[X X_1 \dots X_m]$  is a construction of order  $n$  over  $B$ .
- iv) Let  $x_1, \dots, x_m, X$  ( $m > 0$ ) be constructions of order  $n$  over  $B$ . Then  $[\lambda x_1 \dots x_m X]$  is a construction of order  $n$  over  $B$ .
- v) Nothing is a construction of order  $n$  over  $B$  unless it so follows from  $C_n$  (i)-(iv).

**$T_{n+1}$  (types of order  $n + 1$ )**

Let  $*_n$  be the collection of all constructions of order  $n$  over  $B$ . Then

- i)  $*_n$  and every type of order  $n$  are types of order  $n + 1$ .
- ii) If  $m > 0$  and  $\alpha, \beta_1, \dots, \beta_m$  are types of order

$n + 1$  over  $B$ , then  $(\alpha \beta_1 \dots \beta_m)$  (see  $T_1$  ii)) is a type of order  $n + 1$  over  $B$ .

- iii) Nothing is a type of order  $n + 1$  over  $B$  unless it so follows from (i) and (ii).

Remark. For the purposes of the analysis of our sample example of agents' learning the concept of myopia intensional fragment of TIL based on the simple types of order 1 suffices. Yet when the agents learn new concepts, they enrich their ontology by new constructions that are just displayed rather than executed. To this end, the full ramified hierarchy is needed. For details see, e.g., [7, 8].

For the purposes of natural-language analysis, we are usually assuming the following base of elementary types:

- $\circ$ : the set of truth-values  $\{\mathbf{T}, \mathbf{F}\}$ ;
- $\iota$ : the set of individuals (the universe of discourse);
- $\tau$ : the set of real numbers (doubling as times);
- $\omega$ : the set of logically possible worlds (the logical space).

We model sets and relations by their characteristic functions. Thus, for instance,  $(\circ\iota)$  is the type of a set of individuals, while  $(\circ\iota\iota)$  is the type of a relation-in-extension between individuals. Empirical expressions denote *empirical conditions* that may or may not be satisfied at the world/time pair selected as points of evaluation. We model these empirical conditions as possible-world-semantic *intensions*. Intensions are entities of type  $(\beta\omega)$ : mappings from possible worlds to an arbitrary type  $\beta$ . The type  $\beta$  is frequently the type of the *chronology* of  $\alpha$ -objects, i.e., a mapping of type  $(\alpha\tau)$ . Thus  $\alpha$ -intensions are usually functions of type  $((\alpha\tau)\omega)$ , abbreviated as ' $\alpha_{\tau\omega}$ '. *Extensional entities* are entities of a type  $\alpha$  where  $\alpha \neq (\beta\omega)$  for any type  $\beta$ .

Hence, *empirical expressions* denote (non-trivial, i.e. non-constant) intensions. Where variable  $w$  ranges over  $\omega$  and  $t$  over  $\tau$ , the following logical form essentially characterizes the logical syntax of empirical language:

$$\lambda w \lambda t [\dots w \dots t \dots].$$

Examples of frequently used intensions are:

- propositions of type  $o_{\tau\omega}$  denoted by sentences like “John is a student”;
- properties of individuals of type  $(o\iota)_{\tau\omega}$  denoted by nouns and adjectives, e.g. ‘student’, ‘red’, ‘tall’, ‘myopia’, ‘near-sighted’;
- binary relations-in-intension between individuals of type  $(o\iota\iota)_{\tau\omega}$ , e.g. being ‘composed of’, ‘to like’;
- individual offices (or roles) of type  $\iota_{\tau\omega}$  that are denoted by definite descriptions like ‘the tallest mountain’, ‘Miss World 2019’, ‘the President of Zanzibar’.

Logical objects like *truth-functions* and are extensional:  $\wedge$  (conjunction),  $\vee$  (disjunction) and  $\supset$  (implication) are of type  $(ooo)$ , and  $\neg$  (negation) of type  $(oo)$ . The *quantifiers*  $\forall^\alpha$ ,  $\exists^\alpha$  are type-theoretically polymorphic total functions of type  $(o(o\alpha))$ , for an arbitrary type  $\alpha$ , defined as follows. The *universal quantifier*  $\forall^\alpha$  is a function that associates a class  $A$  of  $\alpha$ -elements with **T** if  $A$  contains all elements of the type  $\alpha$ , otherwise with **F**. The *existential quantifier*  $\exists^\alpha$  is a function that associates a class  $A$  of  $\alpha$ -elements with **T** if  $A$  is a non-empty class, otherwise with **F**.

Notational conventions. Below all type indications will be provided outside the formulae in order not to clutter the notation. The outermost brackets of Closures will be omitted whenever no confusion arises. Furthermore, ‘ $X/\alpha$ ’ means that an object  $X$  is (a member) of type  $\alpha$ . ‘ $X \rightarrow_\nu \alpha$ ’ means that  $X$  is typed to  $\nu$ -construct an object of type  $\alpha$ , regardless of whether  $X$  in fact constructs anything. We write ‘ $X \rightarrow \alpha$ ’ if what is  $\nu$ -constructed does not depend on a valuation  $\nu$ . Throughout, it holds that the variables  $w \rightarrow_\nu \omega$  and  $t \rightarrow_\nu \tau$ . If  $C \rightarrow_\nu \alpha_{\tau\omega}$  then the frequently used Composition  $[[C\ w]\ t]$ , which is the intensional descent (a.k.a. extensionalization) of the  $\alpha$ -intension  $\nu$ -constructed by  $C$ , will be encoded as ‘ $C_{wt}$ ’. When applying quantifiers, we use a simpler notation ‘ $\forall x\ B$ ’, ‘ $\exists x\ B$ ’ instead of the full notation ‘ $[^0\forall^\alpha \lambda x\ B]$ ’, ‘ $[^0\exists^\alpha \lambda x\ B]$ ’,  $x \rightarrow \alpha$ ,  $B \rightarrow o$ , to make the quantified constructions easier to read. When applying truth-functions we use infix notation without Trivialization. For instance, instead of the Composition ‘ $[^0\wedge A\ B]$ ’ we write simply ‘ $[A \wedge B]$ ’.

<sup>4</sup> For details see [8, §4.1]

For illustration, here is an example of the analysis of a simple sentence “John is near-sighted”. First, type-theoretical analysis, i.e. assigning types to the objects that receive mention in the sentence:  $John/\iota$ ;  $Nearsighted/(o\iota)_{\tau\omega}$ ; the whole sentence denotes a proposition of type  $o_{\tau\omega}$ . Now we compose constructions of these objects to construct the denoted proposition. To predicate the property of being near-sighted of John, the property must be extensionalized first:  $[[^0Nearsighted\ w]\ t]$ , or  $^0Nearsighted_{wt} \rightarrow_\nu (o\iota)$ , for short. The Composition  $[^0Nearsighted_{wt}\ ^0John] \rightarrow_\nu o$ ; and finally, the whole empirical sentence denotes a proposition of type  $o_{\tau\omega}$ , hence it encodes as its meaning the Closure

$$\lambda w \lambda t [^0Nearsighted_{wt}\ ^0John] \rightarrow o_{\tau\omega}.$$

In TIL we reject individual essentialism; instead, we adhere to *intensional essentialism*. It means that each  $\alpha$ -intension  $P$  is necessarily related to a collection of *requisites* of  $P$ , its *essence*, that together define the intension  $P$ . For instance, requisites of the property of being a horse are the property of being a mammal of the family Equidae, species *Equus Caballus*, the property of having blood circuit, being a living creature, and many others. Necessarily, if some individual  $a$  happens to be a horse then  $a$  is a mammal of the family Equidae, etc.

The requisite relations *Req* are a family of relations-in-extension between two intensions, hence of the polymorphous type  $(o\alpha_{\tau\omega}\beta_{\tau\omega})$ , where possibly  $\alpha = \beta$ . Infinitely many combinations of *Req* are possible, but the following is the relevant one that we need for our purpose:<sup>4</sup>

$Req/(o(o\iota)_{\tau\omega}(o\iota)_{\tau\omega})$ : an individual property is a requisite of another such property. Thus, we define:

**Definition (requisite relation between  $\iota$ -properties)** Let  $X$ ,  $Y$  be constructions of properties,  $X$ ,  $Y/*_n \rightarrow (o\iota)_{\tau\omega}$ ;  $x \rightarrow \iota$ ,  $True/(oo_{\tau\omega})_{\tau\omega}$ : the property of propositions of being true in a given world and time of evaluation. Then  $[^0Req\ Y\ X] = \forall w \forall t [ \forall x [ [^0True_{wt}\ \lambda w \lambda t [X_{wt}\ x]] \supset [^0True_{wt}\ \lambda w \lambda t [Y_{wt}\ x]] ] ]$ .

Gloss *definiendum* as, “Y is a requisite of X”, and *definiens* as, “Necessarily, at every  $\langle w, t \rangle$ , whatever  $x$  instantiates X at  $\langle w, t \rangle$  also instantiates Y at  $\langle w, t \rangle$ .”

Remark. Here we have to apply the property of propositions *True* to deal with partiality. This is due to the fact that there is a stronger relation between properties, namely that of *pre-requisite*. If Y is a pre-requisite of X, then if an individual  $x$  does not instantiate Y it is neither true nor false that  $x$  instantiates X. The proposition  $\lambda w \lambda t [X_{wt} x]$  has a truth-value gap. For instance, the property of having stopped smoking has a pre-requisite of being an ex-smoker. If somebody never smoked they could not stop smoking, of course. Then, however, the Composition [ ${}^0\text{True}_{wt} \lambda w \lambda t [X_{wt} x]$ ] is simply false and since it is an antecedent of the above implication, the implication is true, as it should be.

Since the topic of this paper is learning and refining concepts, we need to define the notion of concept. In TIL *concepts* are defined as closed constructions in their normal form. Referring for details to [8, §2.2], we briefly recapitulate. Concepts are meanings of semantically complete terms that do not contain indexicals or other pragmatically incomplete terms. In case of the latter we furnish a pragmatically incomplete expression with an open construction containing free variables.

An open construction cannot be executed unless valuation of its free variables is supplied, usually by the situation of utterance. For instance, the meaning of the sentence “He is smart” is the open construction  $\lambda w \lambda t [{}^0\text{Smart}_{wt} he]$ ,  $he \rightarrow_v t$ , that cannot be evaluated until an individual is assigned to the free variable *he* as its valuation.<sup>5</sup> Hence, we don’t treat this open construction as a concept. Since concepts should be at least in principle executable in any state of affairs, we explicate them as closed constructions. However, our TIL constructions are a bit too fine-grained from the procedural point of view.

<sup>5</sup> If such a sentence occurs in a broader discourse, its meaning can be completed by anaphoric references as well. For instance, in “John is a student, he is smart” the meanings are not pragmatically incomplete, because the individual John is substituted for the anaphoric variable *he*. For details on resolving anaphoric references in TIL, see [6].

Some closed constructions differ so slightly that they are *virtually* identical. In a natural language we cannot even render their distinctness, which is caused by the role of  $\lambda$ -bound variables that lack a counterpart in natural languages. These considerations motivated definition of the relation of *procedural isomorphism* on TIL constructions.<sup>6</sup> Procedurally isomorphic constructions form an equivalence class at which we can vote for a representative. To this end a normalization procedure has been defined that results in the unique *normal form* C of a construction that is a representative of the class of procedurally isomorphic constructions. Hence, we adopt this definition:

**Definition (concept)** A *concept* is a closed construction in its normal form.

For the sake of simplicity, in what follows we deal with concepts simply as with closed constructions, ignoring the above technicalities, because we believe that this simplification is harmless for our purposes. The last notion we need to define is that of *refinement* of a concept. Basically, by refining a simple concept  ${}^0O$  of an object  $O$  we mean replacing  ${}^0O$  by an equivalent molecular concept  $C$  that produces the same object  $O$ . We also say that the molecular construction  $C$  is an *ontological definition* of the object  $O$ .

Here is an example. The Trivialization  ${}^0\text{Prime}$  is in fact the least informative procedure for producing the set of prime numbers.

Using particular definitions of the set of primes, we can refine the simple concept  ${}^0\text{Prime}$  in many ways, including:<sup>7</sup>

$$\begin{aligned} & \lambda x [{}^0\text{Card } \lambda y [{}^0\text{Divide } y x] = {}^02], \\ & \lambda x [[x \neq {}^01] \wedge \forall y [[{}^0\text{Divide } y x] \supset \\ & \quad [[y = {}^01] \vee [y = x]]], \\ & \lambda x [[x > {}^01] \wedge \neg \exists y [[y > {}^01] \wedge [y < x] \wedge [{}^0\text{Divide } y x]]. \end{aligned}$$

<sup>6</sup> For details, see [7].

<sup>7</sup> For the sake of simplicity, here we again use infix notation without Trivialization for the application of the binary relations  $>$ ,  $<$  and the identity  $=$  between numbers.

The involved types are:  $v$ , the type of natural numbers;  $Card/(v(ov))$ : the cardinality of a set of natural numbers;  $Divide/(ovv)$ : the relation of  $x$  being divisible by  $y$ ; the other types are obvious. Thus, we define.

**Definition (refinement of a construction)** Let  $C_1, C_2, C_3$  be constructions. Let  ${}^0X$  be a simple concept of an object  $X$  and let  ${}^0X$  occur as a constituent of  $C_1$ . If  $C_2$  differs from  $C_1$  only by containing in lieu of  ${}^0X$  an ontological definition of  $X$ , then  $C_2$  is a *refinement* of  $C_1$ . If  $C_3$  is a refinement of  $C_2$  and  $C_2$  is a refinement of  $C_1$ , then  $C_3$  is a *refinement* of  $C_1$ .

Corollary. If  $C_2$  is a refinement of  $C_1$ , then  $C_1, C_2$  are equivalent but not procedurally isomorphic.

For instance, the simple concept of primes is not procedurally isomorphic with the above refinements, of course, which are molecular concepts with much richer structure than just  ${}^0Prime$ . As a result, the term ‘prime’ is not synonymous with its equivalents like ‘the set of naturals with just two factors’, ‘the set of naturals distinct from 1 that are divisible just by the number 1 and themselves’, because the meanings of *synonymous terms* are procedurally isomorphic. Rather, ‘prime’ is only equivalent to these definitions. So much for our formalism and background theory.

### 3 Supervised Machine Learning

Supervised machine learning is a method of predicting functional dependencies between input values and the output value.

The supervisor provides an agent/learner with a set of training data. These data describe an object by a set of attribute values such that there is a functional dependency between these values.

For instance, a house can be characterized by its size, locality, date of building, architecture style, etc., and its price. Obviously, the price of a house depends on its size, locality, date of building and architecture style. Hence, the price is called an output attribute and the other attributes are input attributes.

The goal of learning is to discover this functional dependency on the grounds of training data

examples so that the agent can predict the value of the output attribute given the values of input attributes of a new instance.

More generally, where  $x_1, \dots, x_n$  are values of input attributes and  $y$  an output attribute value, there is a function  $f$  such that  $y = f(x_1, \dots, x_n)$ . The goal of the learning process is to discover a function  $h$  that approximates the function  $f$  as close as possible. The function  $h$  is called a *hypothesis*. The learner creates hypotheses on the grounds of training data (input-output values) provided by the supervisor.

Correctness of the hypothesis is verified by using a set of test examples given their input attributes. The hypothesis is plausible if the learner predicts the values of the output attribute with a maximum accuracy.<sup>8</sup> Since we decided to apply this method to learning *concepts*, we have to adjust the method a bit. First, instead of input/output attributes, we deal with concepts, that is closed constructions. The role of input ‘attributes’ is played by the constituents of a hypothetic molecular concept and instead of the output attribute we deal with the simple atomic concept that the learner aims to refine.

The hypothetic function is that of a *requisite*. Training data are natural-language texts. The supervisor extracts from the text data positive and negative examples. For instance, let the ‘output’ concept to be learned be that of a cat, i.e.  ${}^0Cat$ . The role of positive examples is played by particular descriptions of the property of being a cat like “Cat is a predatory mammal that has been domesticated”. The learner establishes a hypothesis that the property:

$$\lambda w \lambda t \lambda x [ [ [ {}^0Predatory {}^0Mammal ]_{wt} x ] \wedge [ {}^0Domesticated_{wt} x ] ],$$

belongs to the essence of the property *Cat*. Negative examples delineate the hypothesis from other similar objects. For instance, the sentence “Dog is a domesticated predatory mammal that barks” can serve as a negative example for *Cat*. This triggers a specialization of the hypothetic concept to the construction:

$$\lambda w \lambda t \lambda x [ [ [ {}^0Predatory {}^0Mammal ]_{wt} x ] \wedge [ {}^0Domesticated_{wt} x ] \wedge \neg [ {}^0Bark_{wt} x ] ].$$

<sup>8</sup> For details, see [14, 15, 16].



Hence, given a positive example, the learner refines the hypothetic molecular concept by adding other concepts to the essence, while a negative example triggers specialization of the hypotheses. The hypothetic concept can be also generalized. For instance, the learner can obtain the sentence “Cat is a *wild* feline predatory mammal” as another positive example describing the property *Cat*. Since the properties *Wild* and *Domesticated* are inconsistent, the agent consults his/her ontology for a more general concept. If there is none, the ‘union’ of the properties, *Wild* or *Domesticated*, is included. As a result, the learner obtains this hypothesis:

$$\lambda w \lambda t \lambda x [ [ [ {}^0\text{Feline } [ {}^0\text{Predatory } {}^0\text{Mammal} ] ]_{wt} x ] \wedge [ [ {}^0\text{Domesticated}_{wt} x ] \vee [ {}^0\text{Wild}_{wt} x ] ] \wedge \neg [ [ {}^0\text{Bark}_{wt} x ] ] ].$$

*Remark.* Both *Feline* and *Predatory* are property modifiers of type  $((ot)_{\tau o}(ot)_{\tau o})$ , i.e. functions that given an input property return another property as an output. Since these two modifiers are intersective, the rules of left- and right-subsectivity are applicable here.<sup>9</sup> In other words, predatory mammal is a predator and is a mammal, similarly for feline.

If our agent has these pieces of information in their knowledge base, the above Composition  $[ [ {}^0\text{Feline } [ {}^0\text{Predatory } {}^0\text{Mammal} ] ]_{wt} x ]$  can be further refined to  $[ [ {}^0\text{Feline}'_{wt} x ] \wedge [ {}^0\text{Predatory}'_{wt} x ] \wedge [ {}^0\text{Mammal}_{wt} x ] ]$ , where *Feline'* and *Predatory'* are properties of individuals, i.e. objects of type  $(ot)_{\tau o}$ .

Both generalization, specialization and conjunctive extension are methods of refining a hypothetic concept, the methods that we are going to describe in the next section.

### 3.1 Refining Hypothesis Space

In our method we try to find the description of all plausible hypotheses that are consistent with the training data and are derivable from the provided examples.<sup>10</sup> To this end we assume that there is

<sup>9</sup> For details and analysis of other kinds of modifiers, see [5].

<sup>10</sup> Hypothesis is consistent with the training data, i.e. the set  $S$  of examples, if the value predicted by the hypothesis is the value of output attribute of all examples belonging to  $S$ .

no noise in the training data [14]. In other words, the examples supplied to the learner are adequate for the prediction of the refined concept.

Obviously, a learner can usually examine just a small finite training set of examples instead of a possibly infinite set of sample concepts. Hence, *inductive learning* is applied to obtain a hypothetic concept.<sup>11</sup> In the process of inductive learning, the relation ‘*more general*’ defined on the set of hypotheses is used. This relation is defined as follows. Let  $h_1, h_2$  be hypothetic concepts defined on an input domain  $X$ . Then  $h_1$  is *more general* than  $h_2$ , in symbols ‘ $h_2 \Rightarrow h_1$ ’, iff:

$$\forall x \in X [(h_2(x) = 1) \supset (h_1(x) = 1)].$$

Note. By  $(h_i(x) = 1)$  we mean that an object  $x$  falls under the concept  $h_i$  in a given state of affairs. Hence, this simplified notation can be read as “all objects  $x$  that fall under the concept  $h_2$  fall also under the more general concept  $h_1$ ”. The subset of hypotheses obtained by inductive learning which is consistent with the training set of examples is called *version-space*.

### 3.2 Algorithm Framework

All machine learning algorithms, no matter into which family they belong, can be characterized by common categories which form a framework [11]. The algorithms are characterized by task goals, training data, data representation, and a set of operators which manipulate with data representation. In our machine learning algorithm, the framework can be briefly described as follows.

**Objective Goal.** As mentioned above, the goal of an agent is to discover the best refinement of the learned simple concept of an object  $O$ , i.e. a molecular closed construction that produces the same object. Moreover, this molecular concept should specify as much as possible of the requisites of the object  $O$  so that it also excludes other similar concepts.

**Training data.** An agent works with positive and negative examples that are sentences extracted by

<sup>11</sup> For details on and definition of inductive learning see, e.g., [14, §2.2.2, p. 23].

a supervisor from a textual base. Positive examples contain concepts of requisites specifying the learned simple concept, while negative examples specify properties that do not belong to the essence of the intension provided by the concept.

**Data Representation.** The agents must have an internal formal representation of data obtained by examples. Plausible hypotheses are then formulated in terms of this representation. Our formalism is that of Transparent Intensional Logic so that the sentences are analysed in terms of TIL constructions.

**Knowledge Modifying Module.** The learning algorithm is biased in favor of a preferred hypothesis. By using proper preferences, we reduce the hypothesis space. In version-space learning the bias is called a restriction bias, because the bias is obtained by restricting the allowable hypotheses. The agent uses a set of operations to modify the hypothesis during a *heuristic* search in the hypothesis space. The three main operations to modify a hypothetical concept are generalization, specialization and refinement. There are two possibilities how to obtain a proper hypothesis. The first one is based on using merely positive examples. In this case we need to be sure that the examples cover well the positive cases; in other words, we need examples containing all and only requisites of the learned concept.

The second way that we vote for is using both positive and negative examples. By applying specialization based on negative examples we exclude too general hypotheses.

## 4 Inductive Heuristics

For our purpose we voted for an adjusted version of Patrick Winston algorithm [18] of supervised machine learning. This algorithm applies the principles of generalization and specialization to obtain a plausible hypothesis, i.e. the functional dependency between input and output attributes. In our case the main principle is the method of refining the output simple concept. Hence, instead of a functional dependency between input and output attributes, we are looking for molecular concepts refining the output simple concept; constituents of the molecular concept are related

to the output concept by the requisite relation. Winston algorithm assumes that examples differ from the model just in one attribute while in our case we develop the molecular concept by adding new constituents contained in example sentences describing or rather refining the output concept. Hence our algorithm does not compare a model with examples; rather, it compares the hypothetical concept with information in sample sentences.

As stated above, our main method is *refinement* of a concept, i.e. a hypothetical molecular construction. Based on positive examples we extend the collection of requisites by adding missing concepts in a conjunctive way. As a special case, *generalization* can be applied. Based on agents' ontologies, generalization usually concerns replacing one or more constituents of the hypothetical concept by a more general one.

*Specialization* is triggered by negative examples. As a result, negation of a property that does not belong to the essence of the hypothetical concept is inserted. Specialization serves to distinguish the output concept from similar ones. For instance, a wooden horse can serve as a negative example to the concept of horse, because a wooden horse is not a horse; rather, it is a toy horse though it may look like a genuine living horse.

Heuristic methods of the original Winston algorithm work with examples that cover all the attributes of a learned object. Based on positive examples the hypothesis is modified in such a way that the values of attributes are adjusted, or in case of a negative example an unwanted attribute marked as Must-not-be is inserted.

In our application the sentences that mention the learned concept contain as constituents some but not all the requisites of this concept, and we build up a new molecular concept by adding new information extracted from positive or negative examples. Hence, we had to implement a new heuristic *Concept-introduction* for inserting concepts of new requisites into a hypothetical concept. Negative examples trigger the method *Forbid-link* that inserts a concept of negated property into the hypothesis. Generalization is realized by modules that introduce a concept of a more general property; to this end we also adjusted the original heuristic *Close-interval* so that it is possible to generalize values of numeric concepts

by the union of interval values from an example and model.<sup>12</sup> Here is a brief specification of the algorithm.

### Refinement

1. Compare the model hypothesis (to be refined) and the positive example to find a significant difference
2. If there is a significant difference, then
  - a) if the positive example contains as its constituent a concept that the model does not have, use the **Concept-introduction**
  - b) else ignore example

### Specialization

1. Compare the model hypothesis (to be refined) and the near-miss example to find a significant difference
2. If there is a significant difference, then
  - c) if the near-miss example has a constituent of the concept that the model does not have, use the **Forbid-link**
  - d) else ignore example

### Generalization

1. Compare the model hypothesis (to be refined) and the positive example to determine a difference
2. For each difference do
  - a) if a concept in the model points at a value that differs from the value in the example, then
    - i) if the properties in which the model and example differ have the most specific general property, use the **Climb-tree**
    - ii) else use **Union-set**
  - b) if the model and example differ at an attribute numerical value or interval, use the **Close-interval**
  - c) else ignore example.

<sup>12</sup> For the sake of simplicity, we did not change the original names of particular modules though we do not work with 'links' between objects and attribute values any more. The

## 5 Example of Learning the Concept of Myopia

As a sample example we now introduce the process of learning refinements of the simple concept of myopia, i.e.  ${}^0Myopia$ , by extracting information from natural language sentences describing the property of having myopia. As always, first types  $Myopia / (oi)_{\tau\omega}$  *Sharp, Blur, Disorder, Eye-Nerve, Eye-Lenses* /  $(oi)_{\tau\omega}$  *Eye-Focus, Damaged, Inflexible* /  $((oi)_{\tau\omega}(oi)_{\tau\omega})$  *Close, Distant, Looking-at* /  $(ou)_{\tau\omega}$   $x, y \rightarrow \iota$  *Req* /  $(o(oi)_{\tau\omega}(oi)_{\tau\omega})\exists / (o(oi))$

**Positive examples** trigger the heuristic *Concept-introduction* that inserts a concept of a new requisite into the concept learned so far:

1. In myopia, close objects look sharp:

$$\left[ {}^0Req \left[ \lambda w \lambda t \lambda x \left[ {}^0\exists \lambda y \left[ \left[ {}^0Looking-at_{wt} x y \right] \wedge \left[ {}^0Close_{wt} x y \right] \right] \supset \left[ {}^0Sharp_{wt} y \right] \right] \right] {}^0Myopia \right].$$

2. In myopia, distant objects appear blurred:

$$\left[ {}^0Req \left[ \lambda w \lambda t \lambda x \left[ {}^0\exists \lambda y \left[ \left[ {}^0Looking-at_{wt} x y \right] \wedge \left[ {}^0Distant_{wt} x y \right] \right] \supset \left[ {}^0Blur_{wt} y \right] \right] \right] {}^0Myopia \right].$$

3. Myopia is an eye focusing disorder:

$$\left[ {}^0Req \left[ {}^0Eye-Focus {}^0Disorder \right] {}^0Myopia \right].$$

**Negative examples** trigger the heuristic *Forbid-link* that inserts a negative information to the concept learned so far:

1. Cause of myopia is not a damaged eye-nerve:

heuristics *Require-link* and *Drop-link* from the original algorithm have not been used in our adjusted version.

$\neg [{}^0\text{Cause} [{}^0\text{Damaged } {}^0\text{Eye-Nerve}] {}^0\text{Myopia}]$ .

2. Cause of myopia is not inflexible eye lenses:

$\neg [{}^0\text{Cause} [{}^0\text{Inflexible } {}^0\text{Eye-Lenses}] {}^0\text{Myopia}]$ .

Cause/(o(o1)<sub>τo</sub>(o1)<sub>τo</sub>): relation between properties; for the sake of simplicity we analyse 'cause of something' just as such a relation though we are aware of the fact that the problem of the semantics of causal relations is much more complicated. Yet such a simplification is harmless in our example.

### Simulation of the Algorithm Execution

The algorithm creates a molecular concept that will serve as an explication of the learned simple concept step by step. Each positive/negative example yields conjunctive/disjunctive or negative insertion of new constituents into the hypothesis learned so far. The execution of our algorithm begins with a first chosen positive example.

The construction encoded by this sentence (see above) becomes an *initial* hypothesis model:

*"In myopia, close objects look sharp."*

The second positive example:

*"In myopia distant objects appear blur."*

refines the model by **Concept-introduction**. This heuristic module inserts a new concept into the hypothetic model in the *conjunctive* way. As a result, we have a hypothetic model *"In myopia, close objects look sharp and distant objects look blur"*:

$$\left[ {}^0\text{Req} \left[ \lambda w \lambda t \lambda x \left[ {}^0\exists \lambda y \left[ \left[ {}^0\text{Looking-at}_{wt} x y \right] \wedge \left[ {}^0\text{Close}_{wt} x y \right] \supset \left[ {}^0\text{Sharp}_{wt} y \right] \right] \right] {}^0\text{Myopia} \right] \wedge, \right. \\ \left. \left[ {}^0\text{Req} \left[ \lambda w \lambda t \lambda x \left[ {}^0\exists \lambda y \left[ \left[ {}^0\text{Looking-at}_{wt} x y \right] \wedge \left[ {}^0\text{Distant}_{wt} x y \right] \supset \left[ {}^0\text{Blur}_{wt} y \right] \right] \right] {}^0\text{Myopia} \right] \right]. \right.$$

The last positive example:

*"It is an eye focusing disorder."*

also refines the model by *Concept-introduction*:

$$\left[ {}^0\text{Req} \left[ \lambda w \lambda t \lambda x \left[ {}^0\exists \lambda y \left[ \left[ {}^0\text{Looking-at}_{wt} x y \right] \wedge \left[ {}^0\text{Close}_{wt} x y \right] \supset \left[ {}^0\text{Sharp}_{wt} y \right] \right] \right] {}^0\text{Myopia} \right] \wedge, \right. \\ \left[ {}^0\text{Req} \left[ \lambda w \lambda t \lambda x \left[ {}^0\exists \lambda y \left[ \left[ {}^0\text{Looking-at}_{wt} x y \right] \wedge \left[ {}^0\text{Distant}_{wt} x y \right] \supset \left[ {}^0\text{Blur}_{wt} y \right] \right] \right] {}^0\text{Myopia} \right] \wedge, \right. \\ \left. \left[ {}^0\text{Req} \left[ {}^0\text{Eye-Focus } {}^0\text{Disorder} \right] {}^0\text{Myopia} \right]. \right.$$

The first negative example "The cause of myopia is not a damaged eye nerve" triggers specialization of the hypothesis. As a result, we insert *negative* information about myopia:

$$\left[ {}^0\text{Req} \left[ \lambda w \lambda t \lambda x \left[ {}^0\exists \lambda y \left[ \left[ {}^0\text{Looking-at}_{wt} x y \right] \left[ \quad \right] \right] \right] {}^0\text{Myopia} \right] \wedge, \right. \\ \left[ {}^0\text{Req} \left[ {}^0\text{Eye-Focus } {}^0\text{Disorder} \right] {}^0\text{Myopia} \right] \wedge, \\ \left[ {}^0\neg \left[ {}^0\text{Cause} \left[ {}^0\text{Damaged } {}^0\text{Eye-Nerve} \right] {}^0\text{Myopia} \right] \right].$$

The second negative example "Myopia is not caused by inflexible eye lenses" also specializes the concept. The resulting molecular concept is:

$$\left[ {}^0\text{Req} \left[ \lambda w \lambda t \lambda x \left[ {}^0\exists \lambda y \left[ \left[ {}^0\text{Looking-at}_{wt} x y \right] \wedge \left[ {}^0\text{Close}_{wt} x y \right] \supset \left[ {}^0\text{Sharp}_{wt} y \right] \right] \right] {}^0\text{Myopia} \right] \wedge, \right. \\ \left[ {}^0\text{Req} \left[ \lambda w \lambda t \lambda x \left[ {}^0\exists \lambda y \left[ \left[ {}^0\text{Looking-at}_{wt} x y \right] \wedge \left[ {}^0\text{Distant}_{wt} x y \right] \supset \left[ {}^0\text{Blur}_{wt} y \right] \right] \right] {}^0\text{Myopia} \right] \wedge, \right. \\ \left[ {}^0\text{Req} \left[ {}^0\text{Eye-Focus } {}^0\text{Disorder} \right] {}^0\text{Myopia} \right] \wedge, \\ \left[ {}^0\neg \left[ {}^0\text{Cause} \left[ {}^0\text{Damaged } {}^0\text{Eye-Nerve} \right] {}^0\text{Myopia} \right] \right] \wedge, \\ \left[ {}^0\neg \left[ {}^0\text{Cause} \left[ {}^0\text{Inflexible } {}^0\text{Eye-Lenses} \right] {}^0\text{Myopia} \right] \right].$$

In this example we still did not deal with generalization. *Generalization* triggers one of the three heuristics, namely **Climb-tree**, **Set-union** or **Close-interval**. These heuristic modules adjust the hypothetic concept in the following way.

The **Climb-tree** heuristic module replaces two or more constituents of the concept learned so far by a more general constituent. To this end an agent must have a tree model of organizing concepts in agent's ontology. The concepts in the model are ordered with respect to the requisite relation. For instance, consider the concepts of the properties of having eye correction, wearing dioptric glasses, having contact lenses. Necessarily, if an individual  $x$  happens to have an eye correction then  $x$  wears dioptric glasses or contact lenses, or perhaps has some other eye correction.

Hence, the property of having eye correction is a requisite of both the properties of wearing dioptric glasses and having contact lenses. Let  $Correction/(o1)_{\tau_0}$  be the property of having eye correction,  $Glasses/(o1)_{\tau_0}$  the property of wearing dioptric glasses,  $Lenses/(o1)_{\tau_0}$  the property of having contact lenses.

Then the concept  ${}^0Correction$  is more general than  ${}^0Glasses$  and more general than  ${}^0Lenses$ , i.e. the property *Correction* is a requisite of the properties *Glasses* and *Lenses*.

Hence, assume that a hypothetic model learned so far has a requisite constituent<sup>13</sup>:

$$[{}^0Req \ {}^0Glasses \ {}^0Myopia].$$

Since this construction is too specific to properly explicate Myopia (having myopia, one can have contact lenses or have undergone eye-surgery), assume that the new positive example provided by the supervisor is:

“If somebody has myopia then  
he/she has contact lenses”.

It means that the property  ${}^0Lenses$  should be also inserted as a requisite constituent. However, the

<sup>13</sup> We assume that the supervisor would not stop learning process at this stage, of course, because this hypothesis is too specific and calls for generalizing.

model would be too specific. If somebody has myopia, he/she might have another eye correction.

Therefore, we have to generalize, which is the task that the algorithm performs using agent's ontology. As a result, the new generalized model is adjusted so that the requisite conjunct  $[{}^0Req \ {}^0Glasses \ {}^0Myopia]$  is replaced by the more general conjunct:

$$[{}^0Req \ {}^0Correction \ {}^0Myopia].$$

The **Set-union** heuristic is applied in case we need to generalize several concepts  $C_1, C_2, C_3, \dots$  of properties but there is no most common general concept in agent's ontology. Generalization is achieved by inserting the respective concepts in the *disjunctive way*. For instance, if an agent has a hypothesis that the symptom of myopia is a headache  $[{}^0Symptom \ {}^0Headache \ {}^0Myopia]$  and by a new positive example the agent learns that the symptom of myopia is an eye-ache, the new version of hypothesis will have a constituent that the symptoms of myopia are *Headache* or *Eye-ache*.

$$[{}^0Symptom \ [\lambda w \lambda t \lambda x \ [ [{}^0Headache_{wt} \ x] \vee [{}^0Eye-ache_{wt} \ x]]] \ {}^0Myopia].$$

*Types.*  $Symptom/(o(o1)_{\tau_0}(o1)_{\tau_0})$ : the relation between properties  $P, Q$  such that typically, if  $x$  has the property  $Q$  then  $x$  has also  $P$ ; *Headache, Eye-ache*/( $o1$ ) <sub>$\tau_0$</sub> .<sup>14</sup>

The **Close-interval** heuristic module deals with attributes that have *numerical* values. For instance, in Wikipedia we can read.

The degree of myopia is described in terms of the power of the ideal correction, which is measured in dioptres:

- Low myopia usually describes myopia of −3.00 dioptres or less (i.e. closer to 0.00).
- Moderate myopia usually describes myopia between −3.00 and −6.00 dioptres.

<sup>14</sup> We don't deal with the problem of defining the simple concept  ${}^0Symptom$ . While requisite is a *necessary* relation between properties, symptom is just a *typical* relation between properties.

- High myopia usually describes myopia of -6.00 or more.

To simplify a bit, we just assume that the requisite part of the molecular concept has been inductively enriched by these constituents. Let  $PoC(\tau_i)_{\tau_{10}}$  be the attribute that associates an individual with a number that is the power of correction (measured in dioptres).

**Remark.** Attributes that have numerical values associate an individual with a number. Yet, the situation is a bit more complicated. We also need information on the *unit* in which this number has been obtained. In our example, the agents should know that the power of correction is measured in *dioptres*. Since the issue concerning physical, medical and other units has still not been properly dealt with in TIL, our agents simply keep these pieces of information in their ontology. Hence, when an agent learns that among the requisites of myopia there is the property of having power of correction equal to -5, the agent associates this number with dioptres.

The additional constituents of our hypothetic concept are these.

$$\begin{aligned} & \left[ {}^0Req \left[ \lambda w \lambda t \lambda x \left[ \left[ {}^0 < {}^0-3 \left[ {}^0PoC_{wt} x \right] \right] \right. \right. \right. \\ & \quad \left. \left. \wedge \left[ {}^0 < \left[ {}^0PoC_{wt} x \right] {}^00 \right] \right] \right] {}^0Myopia \right], \\ & \left[ {}^0Req \left[ \lambda w \lambda t \lambda x \left[ \left[ {}^0 < {}^0-6 \left[ {}^0PoC_{wt} x \right] \right] \right. \right. \right. \\ & \quad \left. \left. \wedge \left[ {}^0 < \left[ {}^0PoC_{wt} x \right] {}^0-3 \right] \right] \right] {}^0Myopia \right], \\ & \left[ {}^0Req \left[ \lambda w \lambda t \lambda x \left[ {}^0 < \left[ {}^0PoC_{wt} x \right] {}^0-6 \right] \right] {}^0Myopia \right]. \end{aligned}$$

By applying generalization, i.e. the *Close-interval* heuristic, we obtain a generalized constituent, namely that the power of correction is negative:

$$\left[ {}^0Req \left[ \lambda w \lambda t \lambda x \left[ {}^0 < \left[ {}^0PoC_{wt} x \right] {}^00 \right] \right] {}^0Myopia \right].$$

The resulting explication of ‘myopia’ that our agent has learned is this:

$$\begin{aligned} & \left[ {}^0Req \left[ \lambda w \lambda t \lambda x \left[ {}^0 \exists \lambda y \left[ \left[ \left[ {}^0Looking-at_{wt} x y \right] \right. \right. \right. \right. \\ & \quad \left. \wedge \left[ {}^0Close_{wt} x y \right] \right. \right. \\ & \quad \left. \left. \left. \left. \supset \left[ {}^0Sharp_{wt} y \right] \right] \right] \right] {}^0Myopia \right] \wedge, \\ & \left[ {}^0Req \left[ \lambda w \lambda t \lambda x \left[ {}^0 \exists \lambda y \left[ \left[ \left[ {}^0Looking-at_{wt} x y \right] \right. \right. \right. \right. \right. \\ & \quad \left. \wedge \left[ {}^0Distant_{wt} x y \right] \right. \right. \\ & \quad \left. \left. \left. \left. \supset \left[ {}^0Blur_{wt} y \right] \right] \right] \right] {}^0Myopia \right] \wedge, \\ & \left[ {}^0Req \left[ {}^0Eye-Focus {}^0Disorder \right] {}^0Myopia \right] \wedge, \\ & \left[ {}^0 \neg \left[ {}^0Cause \left[ {}^0Damaged {}^0Eye-Nerve \right] {}^0Myopia \right] \right] \wedge, \\ & \left[ {}^0 \neg \left[ {}^0Cause \left[ {}^0Inflexible {}^0Eye-Lenses \right] {}^0Myopia \right] \right] \wedge, \\ & \left[ {}^0Req {}^0Correction {}^0Myopia \right] \wedge, \\ & \left[ {}^0Symptom \left[ \lambda w \lambda t \lambda x \left[ \left[ {}^0Headache_{wt} x \right] \right. \right. \right. \right. \\ & \quad \left. \left. \vee \left[ {}^0Eye-ache_{wt} x \right] \right] \right] {}^0Myopia \right] \wedge, \\ & \left[ {}^0Req \left[ \lambda w \lambda t \lambda x \left[ {}^0 < \left[ {}^0PoC_{wt} x \right] {}^00 \right] \right] {}^0Myopia \right]. \end{aligned}$$

Obviously, our agent has learned a lot about myopia. Yet, we hesitate to claim that the agent learned a *definition* of the property myopia. The example we presented here is an idealized one.

In practice much more difficulties can crop up. The supervisor could have classified particular constituents improperly, for instance by assigning them the role of a requisite where it might have been just a typical property. Or, the supervisor might have confused symptoms and causes of myopia. Last but not least, pieces of information extracted from the text data might have come from an *unreliable source*. Anyway, we can conclude that the agent discovered a useful *explication* of the simple concept of myopia.

## 6 Conclusion

In this paper we introduced basic principles of supervised machine learning, namely the method of refining hypothesis by means of positive and negative examples.

The process of refinement of a given hypothesis triggered by positive and near-miss examples together with heuristic functions that modify the hypothesis has been described and

illustrated by examples. We applied an adjusted version of Patrick Winston's data driven algorithm for machine learning.

The area under scrutiny has been agents' learning simple concepts by their refining. In other words, our agents learn new concepts by discovering compound concepts that explicate a given simple concept. The method itself has been illustrated by the example of agent's learning the simple concept of myopia. Our data have been formalized by means of the TIL tools, namely constructions and types produced by the NLA algorithm [12].

The proposed machine learning method heavily relies on the role of a supervisor. For a success in learning it is important that the supervisor extracts from a given text those sentences that mention the concept in a way plausible for learning. Moreover, there should not be any noise in these input data, and the supervisor should properly classify these sentences into positive and negative examples and properly recognise those properties that are requisites. Hence, we assume that the role of a supervisor is played by an experienced linguist. As a future research, we intend to extend the functionalities of the algorithm so that it will cover also the extraction of sample sentences where the output learned concept receives mention.

Though there is no substitute for a supervisor in a supervised machine learning method, its role can be at least partly played by the algorithm so that the manual work of a linguist is reduced to a minimum.

Our next goal is to improve the method so that the agents would learn synonymous terms referring to the same concept and distinguish them from merely equivalent ones. This is important for properly dealing with hyperintensional attitudes of knowing, believing, designing, calculating, solving, etc.

These attitudinal verbs are part and parcel of our everyday vernacular so that their proper analysis and logic should not be missing from any automatized multiagent system. And since these attitudinal verbs establish hyperintensional contexts where the substitution of merely equivalent terms fails, the agents need to know the synonyms of the learned concepts as well.

## Acknowledgements

This research has been supported by the Grant Agency of the Czech Republic, project No. GA18-23891S, "Hyperintensional Reasoning over Natural Language Texts" and also by the internal grant agency of VSB-Technical University Ostrava, project No. SP2019/40, "Application of Formal Methods in Knowledge Modelling and Software Engineering II". Versions of this paper were presented at the 20<sup>th</sup> International Conference on Computational Linguistics and Intelligent Text Processing, *CICLing 2019*, France.

## References

1. Carnap, R. (1947). *Meaning and necessity*. Chicago: Chicago University Press.
2. Číhalová, M., Duží, M., & Menšík, M. (2014). Logical Specification of Processes. *Frontiers in Artificial Intelligence and Applications*, Vol. 260, pp. 45–63.
3. Duží, M. (2012). Extensional logic of hyperintensions. *Lecture Notes in Computer Science*, Vol. 7260, pp. 268–290. DOI: 10.1007/978-3-642-28279-9-19.
4. Duží, M. (2014). Communication in a Multi-Cultural World. *Organon F.*, Vol. 21, No. 2, pp. 198–218.
5. Duží, M. (2017). Property modifiers and intensional essentialism. *Computación y Sistemas*, Vol. 21, No. 4, 2017, pp. 601–613. DOI: 10.13053/CyS-21-4-2811.
6. Duží, M. (2018). Logic of Dynamic Discourse; Anaphora Resolution. *Frontiers in Artificial Intelligence and Applications*, Vol. 301, pp. 263–279. DOI 10.3233/978-1-61499-834-1-263.
7. Duží, M. (2019). If structured propositions are logical procedures then how are procedures individuated? *Synthese* special issue on the Unity of propositions, Vol. 196, No. 4, pp. 1249–1283. DOI: 10.1007/s11229-017-1595-5.
8. Duží, M., Jespersen, B., & Materna, P. (2010). Procedural Semantics for Hyperintensional Logic. *Foundations and Applications of Transparent Intensional Logic*.
9. Francez, N. (2015). *Proof-theoretic Semantics*. Studies in Logic 57, College Publications.

10. Kovář, V., Baisa, V., & Jakubíček, M. (2016). Sketch Engine for Bilingual Lexicography. *International Journal of Lexicography*, Vol. 29, No. 3, pp. 339–352.
11. Luger, G. F. (2009). *Artificial intelligence: structures and strategies for complex problem solving*.
12. Medved', M., Šulganová, T., & Horák, A. (2017). Multilinguality Adaptations of Natural Language Logical Analyzer. *Proceedings of the 11<sup>th</sup> Workshop on Recent Advances in Slavonic Natural Language (RASLAN'17)*, pp. 51–58.
13. Menšík, M., Duží, M., Albert, A., & Patschka, V., Pajr, M. (2020). Machine learning using TIL. *To appear in Frontiers in Artificial Intelligence and Applications*.
14. Mitchell, T.M. (1997). *Machine Learning*. McGraw- Hill.
15. Poole, D.L. & Mackworth, A.K. (2010). *Artificial Intelligence: Foundations of Computational Agents*.
16. Russell, S.J. & Norvig, P. (2014). *Artificial Intelligence: a Modern Approach*. Pearson Education.
17. Tichý, P. (1988). *The Foundations of Frege's Logic*.
18. Winston, P.H. (1992). *Artificial Intelligence*. Addison-Wesley Pub. Co.

Article received on 18/01/2019; accepted on 21/02/2019.  
Corresponding author is Marek Menšík.



# Relation between Titles and Keywords in Japanese Academic Papers using Quantitative Analysis and Machine Learning

Masaki Murata, Natsumi Morimoto

Tottori University, Faculty of Engineering, Tottori,  
Japan

{murata, s112057}@ike.tottori-u.ac.jp

**Abstract.** In this study, we analyzed keywords from different academic papers using data from more than 300 papers. Using the concept of quantitative surveys and machine learning, we conducted various analyses on the keywords in different papers. The findings obtained from these surveys and analyses are assumed to lend themselves to the automatic assignment of keywords for papers. In this study, the number of keywords included in a paper is quantitatively expressed using the covering rate and density of keywords. The results confirm that paper titles are likely to include keywords. The performed keyword analyses predict words that can be used as keywords via machine learning. The proposed method has an accuracy range 0.6–0.8. In addition, by analyzing the features used in machine learning, we can obtain the characteristics of the words that are mentioned as keywords in papers.

**Keywords.** Thesis, title, keyword, machine learning, feature analysis

## 1 Introduction

In recent years, along with the enormous increase in the number of electronic documents, the importance of developing a method to facilitate the retrieval of information from large amounts of documents has increased. Many journals require authors to provide keywords for papers.

Nagao [5] proposed a search method that uses the table-of-contents information of books and documents. Nagao mentioned that for academic papers, the number of keywords attached by the author does not sufficiently reflect the content of the paper.

He stated that because the titles of papers and the titles of chapters and sections have an appropriate number of technical terms suitable for expressing the content of a paper, it is reasonable that these titles be used as paper keywords.

Nagao used 16 papers for their analysis. Conversely, in this study, we expand the study scale by using a large amount of data, i.e., 300 papers or more, and primarily analyze the keywords of the papers. We investigate whether the titles in a paper can be used as keywords of the paper and examine the characteristics of the keywords.

The obtained findings will likely lead to the automatic generation of keywords for papers and an improvement in the keyword generation performance.

The highlights of this study can be summarized as follows:

- In this study, we conducted a survey and analysis of keywords in academic papers using a large amount of data. The covering rate of how much the title of a paper covers the keywords given by the author and the density of the keywords the author provided in the title of the paper are investigated. The fact that paper titles tend to include keywords was confirmed.
- We conducted experiments predicting the words that could be keywords of a paper using the titles of papers, its chapters, and its sections via supervised machine learning; our proposed methods obtained

an accuracy rate of 0.6 for the prediction of keywords. In other experiments wherein human beings determined the correct answers in an evaluation, our proposed methods obtained an accuracy rate of 0.8 for the prediction of keywords.

- By analyzing the features used for supervised machine learning, we easily obtained the characteristics of words including “analysis” and found that words in titles tend to be keywords and that words such as “morphological analysis” and “syntactic analysis” tend to be keywords.

## 2 Previous Studies

Nagao [5] proposed a method to select keywords from the table-of-contents information and to search books and documents using the structures of the chapters and sections in the table of contents when creating a retrieval system for books and documents.

He manually investigated 16 academic papers. As a result, for academic papers, he concluded that the number of keywords attached by the authors is inadequate and does not fully reflect the contents of a paper. He stated that because the titles of papers and the titles of chapters and sections include a sufficient number of technical terms to express a paper, it is reasonable for these titles to be used as the keywords of a paper.

Kurohashi et al. [2] examined the strength of the relations among terms in the text and the table of contents using the hierarchically structured information in the table of contents and made it possible to conduct a document search based on not only the conventional AND and OR connections but also the degree of keyword connections. In an evaluation experiment to investigate the effectiveness of these methods, they assessed approximately 17,000 volumes of table-of-contents information. As a result, they quantitatively proved that their proposed method was effective in book searches by evaluating 80 search results by subject.

Nagao and Kurohashi et al. showed that the table-of-contents information can effectively serve

as keywords in book searches. In this study, we also investigated whether the table-of-contents information represents the characteristics of academic papers but used a method different from those of Nagao and Kurohashi et al.

Uchiyama et al. [7] proposed a method to extract keywords from abstracts via statistical methods, using keywords given by the authors of specialty papers as learning data for the keywords. Uchiyama et al.’s proposed method obtained a recall of 0.8 and a precision of 0.43, indicating the effectiveness of their method. Uchiyama et al. extracted keywords given by authors from summaries using statistics. Conversely, in this study, we primarily aim to predict keywords from titles using supervised machine learning. In addition, Uchiyama et al. used the words immediately before and after the target word to obtain statistics for the learning data. Conversely, in this study, learning data are created by using not only words immediately before and after a target word but also the information contained in additional words further from the target.

In addition to the above, many studies concerning keyword extraction have been conducted. Research concerning keywords such as titles and abstracts have also been conducted [1]. In this study, machine learning is used to determine keywords using the frequency of a word, the importance of the sentence in which the word exists, and the length of the word. In our study, the words before and after a word as well as a thesaurus are used as information for the learning process and a machine learning method is used to predict the keywords.

In addition, there are studies [8] that have extracted keywords from documents using multiple machine learning techniques. In our study, machine learning is used as well; however, the keywords are estimated not only from the entire document but also from the table-of-contents information using paper titles and their chapter and section titles, and an analysis of the features is performed. In our study, we focused on the importance of titles.

### 3 How to Proceed with this Research

This study was conducted in the following order:

- Quantitative survey of keywords

We quantitatively described how many keywords were included in a paper using the covering ratio and the density of keywords and conducted surveys on the keywords.

- Analysis of keywords based on machine learning

By predicting which words could become keywords using supervised machine learning techniques such as the maximum entropy method and support vector machine *SVM* and analyzing the features that were useful for the machine-learning-based prediction, it was possible to evaluate the characteristics of words that are likely or unlikely to be keywords.

### 4 Quantitative Survey of Keywords

As mentioned in Section 2, Nagao [5] determined that the keywords given by an author do not adequately reflect the content of a paper. However, the keywords given by an author (which we call author keywords) are important words in the paper and can be used as a certain index for the keywords.

We compared the author keywords in a paper with the following elements of the paper and investigated how many keywords were included in the paper or in each part of the paper:

- Paper title,
- Titles of chapters and sections,
- Titles of papers, chapters, and sections,
- Abstract,
- Titles of papers, chapters, and sections and abstract,
- Entire paper.

#### 4.1 Used Data

In this study, we used 343 papers in the Japanese Journal of Natural Language Processing (over a period of 16 years) as experimental data.

A paper in this dataset contains a title, an abstract, keywords, chapter/section titles, and the text. Each paper has approximately five keywords.

#### 4.2 Survey Method

The covering ratio of author keywords is defined as the number of phrases in an element matching the author keywords divided by the number of author keywords. This ratio was examined for each article, and the average value of the ratio was obtained.

The covering ratio of words in the author keywords is defined as the number of words in an element matching the author keywords divided by the total number of words in the author keywords. This ratio was examined for each article, and the average value of the ratio was obtained. To partition an author keyword into words, we used *ChaSen*<sup>1</sup>.

The density of words in the author keywords is defined as the number of words in an element matching the words in the author keywords divided by the number of words in the element. This ratio was examined for each paper, and the average value of the ratio was obtained.

An example of calculating the covering ratio of the author keywords, the covering ratio of the words in the author keywords, and the density of words is shown in Fig. 1.

#### 4.3 Investigation Result

Table 1 shows the results for the covering ratio of author keywords. Table 2 shows the results for the covering ratio of the words in author keywords. Table 3 shows the results for the density of words in author keywords.

<sup>1</sup><http://chasen-leagacy.sourceforge.jp/>

1. Author keywords  
Machine learning, case analysis, and support vector machine
2. Title  
Case analysis using machine learning
3. Covering ratio of author keywords  
 $2$  (the number of matches) /  $3$  (the number of author keywords)
4. Covering ratio of the words in author keywords  
 $4$  (the number of matches) /  $7$  (the number of author keywords)
5. Density of words  
 $4$  (the number of matches) /  $5$  (the number of words in the title)

**Fig. 1.** Examples of the calculations of covering ratios and density

**Table 1.** Results for the covering ratio of author keywords

Element	Results
Title	0.36
Chapter/Section titles	0.48
Title and chapter/section titles	0.58
Abstract	0.60
Title, chapter/section titles, and abstract	0.71
Entire paper	0.86

#### 4.4 Discussion

First, we numerically compared the results of the covering ratio of author keywords with those of the words in the author keywords. Looking at Tables 1 and 2, the covering ratio of the words in author keywords is approximately 0.2 higher than the covering ratio of author keywords. In the case of the author keywords as is (Table 1), the covering ratio of the titles is approximately 0.4, the covering ratio of the titles and chapter/section titles is approximately 0.5, and the covering ratio of entire papers is approximately 0.9. However, in the case of splitting author keywords into words (Table 2), the covering ratio of the titles is approximately 0.5, the covering ratio of the titles and chapter/section titles is approximately 0.7, and the covering ratio of entire papers is approximately 1.0.

Next, when analyzing the results of the covering ratio of the words in author keywords, words

**Table 2.** Results for the covering ratio of the words in author keywords

Element	Results
Title	0.53
Chapter/Section titles	0.67
Title and chapter/section titles	0.76
Abstract	0.82
Title, chapter/section titles, and abstract	0.89
Entire paper	0.98

**Table 3.** Results for the density of the words in author keywords

Elements	Results
Title	0.41
Chapter/Section titles	0.22
Title and chapter/section titles	0.25
Abstract	0.13
Title, chapter/section titles, and abstract	0.16
Entire paper	0.084

that are not keywords themselves were found to be included in the keywords. For example, we considered the keyword *LR koubun kaiseki* “LR syntax analysis.” Even though this keyword may not appear in an entire given paper, the individual word groupings *LR* “LR” and *koubun kaiseki* “syntax analysis” do appear in such a paper.

Finally, we analyzed the results of the density of words in the author keywords. From Table 3, the density of the entire paper is less than 0.1; however, the density of the titles exceeds 0.4, which is relatively high. The second highest density is that of the titles and chapter/section titles. If words are randomly extracted from an entire paper, the accuracy rate of extracting the correct words from the author keywords is approximately 0.1. However, if words are randomly extracted from a paper title, the accuracy rate of extracting correct words from the author keywords is approximately 0.4. Words from a paper title are therefore more likely to be words in the author keywords than words from the entire paper.

From the above, it can be concluded that words in paper titles or chapter/section titles tend to be keywords.

## 5 Analysis of Keywords based on Machine Learning

Section 4 demonstrated that words that could be keywords were contained in the paper titles. However, unnecessary words that are not keywords are included in titles. Machine learning was performed to judge whether the detected words could be keywords of a given paper. We analyzed the differences between the words that could be keywords and the words that could not.

### 5.1 Our Proposed Method

In this study, we used supervised machine learning to predict which words could be used as keywords of articles. We separate the elements of the paper (i.e., the paper title, chapter title, section title, abstract, and whole paper) into words via *ChaSen* and input them one by one into the machine. The machine judges whether the input word can be a keyword and outputs the result.

In this study, SVM and the maximum entropy method were used for machine learning in our experiments. SVM supports high-performance supervised machine learning and has been used in many studies. The maximum entropy method automatically finds features with high weights, with features easier to analyze than those of SVM. Thus, it was also used for our experiments.

The features used in the machine learning are shown in Table 4. The features that use *Bunrui Goi Hyou* (a Japanese thesaurus) [6] are defined by consulting Murata's method [3] and can utilize semantic categories (e.g., human beings and animals).

### 5.2 Baseline Methods

We explain the baseline methods that were used in the experimental comparison in this section.

#### A Method using all Words as Keywords

In this study, a morphological analysis was performed using *ChaSen* and the elements of the papers (i.e., paper title, chapter/section title, abstract, and entire article) were divided into words. The method “a method using all words as keywords” considers all words as keywords and

**Table 4.** Features used in the machine learning

ID	Feature
1	Element where the target word appears (title, chapter title, section title, abstract, and text)
2	The target word
3	One word just before the target word
4	Two words just before the target word
5	Three words just before the target word
6	One word just after the target word
7	Two words just after target word
8	Three words just after target word
9	Chapter number and section number
10	The three first digits in <i>Bunrui Goi Hyou</i> (a Japanese thesaurus)
11	The five first digits in <i>Bunrui Goi Hyou</i>

was used as one of the comparative methods. This method corresponds to a case where there is no information (features), and it was used to evaluate whether the machine learning technique works well.

#### A Method using all nouns as Keywords

The elements of the papers (i.e., paper title, chapter/section title, abstract, and entire article) were divided into words. The method “a method using all nouns as keywords” considers all nouns as keywords and was also used as a comparative method.

#### Gensen Web

We extracted the keywords of a paper using an automatic technical term extraction service called “Gensen Web.”<sup>2</sup>

In Gensen Web, when we enter a text, the technical terms (keywords) are displayed in descending order of importance. We input 343 papers and obtained the keywords for each paper. We picked out the words appearing in the elements of the papers from the keywords obtained by Gensen Web.

We sorted the words appearing in the elements obtained by Gensen Web in descending order of importance. We selected the top 7 words for the paper title; the top 8 words for the chapter/section title; the top 9 words for the paper title and chapter/section title; the top 12 words for the

<sup>2</sup><http://gensen.dl.itc.u-tokyo.ac.jp/gensenweb.html>

**Table 5.** Number of data items in the training data

Element	Number of papers	Number of target words	Number of words in author keywords	Number of words not in author keywords
Title	171	1628	814	814
Paper/ chapter/ section title	171	2540	1270	1270
Paper/ chapter/ section title and abstract	172	3080	1540	1540

**Table 6.** Number of data items in the test data

Element	Number of papers	Number of target words	Number of words in author keywords	Number of words not in author keywords
Title	172	2063	775	1288
Paper/ chapter/ section title	172	10577	1183	9394
Paper/ chapter/ section title and abstract	172	23967	1438	22529

abstract; the top 13 words for the paper title, chapter/section title, and abstract; and the top 11 words for the entire paper. We used the selected words as the words that Gensen Web considered as words in keywords.

For the number of words we selected, we used the number of words for which we obtained the highest F-measure in the training data.

### The TF–IDF method

In the term frequency–inverse document frequency (TF–IDF) method, we found the TF–IDF value of a word appearing in a paper and extracted words with high TF–IDF values as keywords.

The formula for the TF–IDF method is shown below:

$$tfidf_{i,j} = tf_{i,j} \times idf_i, \quad (1)$$

$$tf_{i,j} = \frac{n_{i,j}}{\sum_k n_{k,j}}, \quad (2)$$

$$idf_i = \log \frac{|D|}{|d : d \ni t_i|}, \quad (3)$$

here,  $n_{i,j}$  is the number of occurrences of a word  $i$  in the document  $j$ ,  $|D|$  is the total number of documents (papers), and  $|\{d : d \ni t_i\}|$  is the number of documents containing the word  $i$ .

We found the TF–IDF values of the words in the 344 papers using the above equation. We extracted the words appearing in the elements of each paper and sorted them in descending order according to their TF–IDF values. We selected the top 8 words for the paper title; the top 13 words for the chapter/section title; the top 12 words for the paper title and chapter/section title; the top 28 words for the abstract; the top 37 words for the

**Table 7.** Results for extracting keywords from paper titles

Element	Recall rate	Precision rate	F-measure
SVM	0.76 (591/775)	0.57(591/1042)	0.65
Maximum entropy	0.72(560/ 775)	0.57 (560/977)	0.64
All words as keywords	1.00(775/775)	0.38(775/2063)	0.55
All nouns as keywords	0.85 (658/775)	0.50(658/1316)	0.63
Gensen Web	0.85 (655/775)	0.57(655/1141)	0.68
TF-IDF method	0.75 (584/775)	0.43(584/1354)	0.55

**Table 8.** Results for extracting keywords from paper/chapter/section titles

Element	Recall rate	Precision rate	F-measure
SVM	0.77 (905/1183)	0.35 (905/2582)	0.48
Maximum entropy	0.77 (913/1183)	0.32 (913/2821)	0.46
All words as keywords	1.00(1183/1183)	0.11(1183/10577)	0.20
All nouns as keywords	0.72 (850/1183)	0.12 (850/6900)	0.21
Gensen Web	0.57 (674/1183)	0.43 (674/1557)	0.49
TF-IDF method	0.32 (384/1183)	0.18 (384/2076)	0.24

paper title, chapter/section title, and abstract; and the top 41 words for the entire paper. We used the selected words as the words that the TF-IDF method considered to be keywords.

For the number of words we selected, we used the number of words for which we obtained the highest F-measure in the training data.

### 5.3 Experimental Data

We used the data described in Section 4.1. The training data used in the experiment are shown in Table 5, and the test data are shown in Table 6.

With respect to the training data, words not included in the keywords were randomly extracted so that the number of words was the same as the number of words included in the keywords. This was done for the following reasons. If the number of words included in the keywords and the number of words not included in the keywords are very different, there is a possibility that the machine learning will not work well. Therefore, in this study, we prevented this problem by using the same quantity for the number of words included in the keywords and that of words not included in the keywords.

### 5.4 Experimental Results

The results obtained using the six methods (our proposed methods (machine learning based on SVM and the maximum entropy method), the method judging all entered words to be keywords, the baseline method, Gensen Web, and the TF-IDF method) are shown in Tables 7–9. The recall rates, precision rates, and F-measures for extracting keywords from the titles are shown in Table 7, the results for the paper/chapter/section titles are shown in Table 8, and the results for the paper/chapter/section titles and abstracts are shown in Table 9. The recall rates indicate the ratio of correct words among words appearing in both the given titles and author keywords, and the precision rates of the correct words among the words that are considered by a given method as words in author keywords. The correct words are the words included in the author keywords.

From Tables 7–9, because the F-measures of the proposed methods are higher than those of the method extracting all words as keywords, we determined that the machine learning method worked well for keyword prediction.

Table 7 indicates that there were smaller differences when comparing the method using all nouns as keywords and the proposed method. However, Tables 7 and 9 indicate that the proposed

**Table 9.** Results for extracting keywords from paper/chapter/section titles and abstracts

Element	Recall rate	Precision rate	F-measure
SVM	0.74(1059/1438)	0.30 (1059/3498)	0.43
Maximum entropy	0.78(1122/1438)	0.28 (1122/4029)	0.41
All words as keywords	1.00(1438/1438)	0.06(1438/23967)	0.11
All nouns as keywords	0.84(1213/1438)	0.09(1213/13979)	0.16
Gensen Web	0.55 (794/1438)	0.35 (794/2249)	0.43
TF-IDF method	0.42 (611/1438)	0.10 (611/6377)	0.16

**Table 10.** Experimental results using correct answers based on three human beings (paper titles)

Method	Recall rate	Precision rate	F-measure
SVM	0.83(19/23)	0.80 (19/(19+3×1288/775))	0.81
Maximum entropy	0.83(19/23)	0.80 (19/(19+3×1288/775))	0.81
All words as keywords	1.0 (23/23)	0.45(23/(23+17×1288/775))	0.62
All nouns as keywords	0.96(22/23)	0.69 (22/(22+6×1288/775))	0.80
Gensen Web	0.70(16/23)	0.76 (16/(16+3×1288/775))	0.73
TF-IDF method	0.70(16/23)	0.52 (16/(16+9×1288/755))	0.59

method obtained higher F-measures than the method extracting all nouns as keywords. Because the recall rate of the proposed method is higher than that of the method using all nouns as keywords, it is likely that the proposed method can distinguish between words that are keywords and words that are not keywords even for non-nouns. Comparing the F-measures of Gensen Web and those of our proposed methods, the values of our proposed method were nearly equivalent to or a little lower than those of Gensen Web. Gensen Web is popular and is often used in many studies. Because our proposed methods obtained values similar to those of Gensen Web, we confirmed that our methods also work well.

Moreover, when we compared the results of the TF-IDF method to those of the proposed method, we found that the proposed method can remove unnecessary words with higher efficiency.

### 5.5 Experiments using Human beings for the Evaluation

We examined the experimental results in the previous section and found that there were words that could be used as keywords other than those provided by the authors. Therefore, we conducted

experiments using keywords determined by human beings other than the authors.

We randomly picked five papers from the test data used in Section 5.3. We randomly extracted 20 words in author keywords and 20 words from the titles (the paper/chapter/section titles) of the papers. Three human beings read the five papers and guessed whether each of the 40 words was included in the author keywords.<sup>3</sup> A word that two or more people considered to be a keyword was regarded as a correct answer.

We evaluated the six methods using the correct answers based on the evaluation of the three human beings mentioned above. The results are shown in Tables 10 and 11. We calculated the recall rates using the sampling data and the following equation:

$$\frac{Sys_1}{Sys_1 + Sys_2 \times R}, \quad (4)$$

here,  $Sys_1$  is the number of correct answers among the words that a system considered to be keywords,  $Sys_2$  is the number of incorrect answers

<sup>3</sup>The kappa coefficient in the manual estimation of the three human beings was 0.4. This corresponds to “moderate agreement.”



**Table 11.** Experimental results using correct answers based on three human beings (paper/chapter/section titles)

Method	Recall rate	Precision rate	F-measure
SVM	0.83(19/23)	0.44 (19/(19+3×9394/1183))	0.58
Maximum entropy	0.83(19/23)	0.37 (19/(19+4×9394/1183))	0.52
All words as keywords	1.0 (23/23)	0.15(23/(23+17×9394/1183))	0.25
All nouns as keywords	0.96(22/23)	0.26 (22/(22+8×9394/1183))	0.41
Gensen Web	0.43(10/23)	0.56 (10/(10+1×9394/1183))	0.49
TF-IDF method	0.30 (7/23)	0.31 (7/(7+2×9394/1183))	0.31

**Table 12.** Results of feature analysis (paper/chapter/section title)

Useful		Not useful	
Feature	Score	Feature	Score
The input word appears in titles	0.93	The succeeding word is “study”	0.15
<i>kaiseki</i> (analysis)	0.86	The previous word is object case particle	0.14
<i>ka</i> (-ization)	0.85	The succeeding word is “method”	0.11

among the words that a system considered to be keywords, and  $R$  is the result of dividing the number of words that are not in author keywords by the number of words that are in author keywords in all the test data.

In the results of Tables 10 and 11, we see that our proposed methods (SVM and maximum entropy) obtained higher F-measures (0.8 for the paper titles) than other methods.

## 5.6 Feature Analysis

In the maximum entropy method, the weights of features are automatically recognized [4]. That is, it is possible to analyze which feature is useful for determining if a word could be a keyword. A feature analysis of the experimental results obtained by the maximum entropy method in Section 5.4 was performed.

Useful features and non-useful features are shown in Table 12. The results were obtained for the experiments including the paper/chapter/title sections. The score in the table indicates the weighting of a feature. A feature with a high score is more useful for predicting keywords.

From Table 12, we found that when an input word exists in a paper title, it tends to be a keyword.

Further, when the input word is “analysis,” it is often a keyword. The terms “morphological

analysis,” “syntax analysis,” “relationship analysis,” etc., were determined to be keywords in the academic field of natural language processing.

We also found that when the word after an input word is “method” or “study,” it does not tend to be a keyword. Accordingly, terms such as “experiment method,” “evaluation method,” “previous study,” etc., are often used for chapters/section titles but are not useful as keywords.

Our study found many other similar keyword characteristics.

## 6 Conclusions

In this study, we investigated and analyzed keywords in academic papers.

The covering rate of how much the title of a paper covers the keywords given by the author and the density of how many keywords the author gave in the title of the paper were investigated. The fact that the paper titles tended to be keywords was confirmed.

Specifically, paper titles covered approximately 0.5 of the keywords given by the author and the chapter/section titles included more than 0.7. Moreover, the density of the paper titles was 0.4 and it was found that keywords can be obtained

with a correct answer rate of 0.4 even if words are randomly selected from the paper title.

We also predicted words that could be the keywords of a paper using supervised machine learning from the various elements of the paper. As a result of keyword prediction from paper titles and paper/chapter/section titles, it was possible to determine words that could be keywords with accuracy rates of 0.6–0.8.

In addition, by analyzing the features used for supervised machine learning, we were able to obtain the characteristics of words that could be keywords of papers.

Specifically, one characteristic of a word that can be a keyword is that it exists in a paper title. It was found that phrases including “analysis,” such as “morpheme analysis” and “syntax analysis” in natural language processing research fields, are likely to be keywords. Conversely, it was found that phrases including “method” and “study,” such as “evaluation method” and “previous study,” are not likely to be keywords.

## References

1. Bhowmik, R. (2008). Keyword extraction from abstracts and titles. *Proceedings of IEEE SoutheastCon 2008*, pp. 610–617.
2. Kurohashi, S., Hagiwara, N., & Nagao, M. (1997). A system for document retrieval by utilizing the table of contents information. *Information Processing Society of Japan, WGFI*, pp. 27–32. (in Japanese).
3. Murata, M. & Isahara, H. (2003). Conversion of Japanese passive/causative sentences into active sentences using machine learning. *Computational Linguistics and Intelligent Text Processing, Second International Conference, CICLing 2003, Mexico City, February 2003 Proceedings*, pp. 115–125.
4. Murata, M., Uchimoto, K., Utiyama, M., Ma, Q., Nishimura, R., Watanabe, Y., Doi, K., & Torisawa, K. (2010). Using the maximum entropy method for natural language processing: Category estimation, feature extraction, and error correction. *Cognitive Computation*, Vol. 2, No. 4, pp. 272–279.
5. Nagao, M. (1992). A method of document retrieval by utilizing the table of contents as keys. *Information Science and Technology Association*, Vol. 42, No. 8, pp. 711–718. (in Japanese).
6. NLRI (1964). *Bunrui Goi Hyou*. Shuuei Publishing.
7. Utiyama, M., Murata, M., & Isahara, H. (2000). Using author keywords for automatic term recognition. *Terminology*, Vol. 6, No. 2, pp. 313–326.
8. Zhang, C. (2009). Combining statistical machine learning models to extract keywords from Chinese documents. *Proceedings of the 5th International Conference on Advanced Data Mining and Applications*, pp. 745–754.

Article received on 17/01/2019; accepted on 06/03/2019.  
Corresponding author is Masaki Murata.

# RelEmb: A Relevance-based Application Embedding for Mobile App Retrieval and Categorization

Shubham Krishna\*, Ahsaas Bajaj\*, Mukund Rungta, Hemant Tiwari, Vanraj Vala

Samsung R&D Institute, Bengaluru,  
India

{shubham.k1, ahsaas.bajaj, mukund.r, h.tiwari, vanraj.vala}@samsung.com

**Abstract.** Information Retrieval Systems have revolutionized the organization and extraction of Information. In recent years, mobile applications (apps) have become primary tools of collecting and disseminating information. However, limited research is available on how to retrieve and organize mobile apps on users' devices. In this paper, authors propose a novel method to estimate app-embeddings which are then applied to tasks like app clustering, classification, and retrieval. Usage of app-embedding for query expansion, nearest neighbor analysis enables unique and interesting use cases to enhance end-user experience with mobile apps.

**Keywords.** Information systems and retrieval, mobile applications, application embedding.

## 1 Introduction

Recent years have seen tremendous increase in usage of mobile applications, a.k.a. *apps*, mainly due to the ever rising popularity of smart phones. There are millions of apps [26] freely available on Google Play Store and Apple App Store. With the abundance of data available in form of applications, it is important to build efficient and effective retrieval engines around them. When a user queries for an application, it is crucial to bring the most relevant application at the top. In a mobile environment, the user expects search operation to be rapid and relevant. This is a challenge as the number of available applications are readily growing these days. Most of the recent work in the field of Information Retrieval has been focused on web-search scenarios and

improving the ranking of web results. Little work is focused on information retrieval centered around mobile applications. Intelligent retrieval methods are required to make sense of this large amount of app data and also keep them organized in users' devices. Since most of the data related to applications are in the form of its description, it is important to mine this source of information. Very recently, neural word embedding has found its use in the field of Information Retrieval. A novel method of learning word embedding from app description is proposed in this paper. The paper is organized as follows.

An overview of related work in the domain of information retrieval is presented in Section 2. The details of estimating the embeddings are discussed in section 3. Section 4 elaborates experiments and results using the proposed embeddings for various tasks. Finally section 5 recapitulates the proposed approach with current applications and scope of future extensions.

## 2 Related Work

In past few years, word embeddings have found a major role for solving different tasks in multiple domains like Natural Language Processing and Machine Learning. Word embedding is basically a representation of a word in a vector space and there are different ways to learn this representation- to model semantic or syntagmatic relationships. Traditionally, techniques like Singular Value Decomposition (SVD) and Principal Component Analysis (PCA) were applied to

\* indicates equal contribution of the authors.

generate dense word representations. With the advent of neural networks, it became possible to learn more enhanced word representations. Different variants of word embeddings such as word2vec [16], GloVe [20] have been proposed for learning dense distributed representation of words. These window based methods count the number of co-occurrences of terms in the training corpus and suggest how likely is a term to occur with other terms. Such methods have been highly effective in solving different problems such as Text Summarization, Sentiment Analysis, and Machine Translation. Later, the idea of neural word embeddings was extended to learn document-level embeddings [13] which opened further interesting use-cases. Information Retrieval is one such domain.

In 2015, a unified framework for monolingual and cross lingual information retrieval was formulated using word embeddings [28]. Dual Embedding Space model using word embeddings was proposed for the task of document ranking [17]. Also, there are methods for solving the problem of term weighting and query interpretation using distributed representation of words [31]. Word and document embeddings have also found their use in applications such as query expansion and estimation of accurate query language models in the task of retrieval [29]. In general, query expansion is based on pseudo relevance feedback [22] to enhance the performance of ad-hoc retrieval. Initial work in query expansion utilized word2vec (continuous bag-of-words) embedding approach [12].

Currently, researchers are experimenting to build robust word embeddings to analyze their usage in different problems of information retrieval. The novelty of locally-trained word embeddings are nicely highlighted in [7]. They have shown that these locally trained word embeddings outperform the globally trained ones (like word2vec, GloVe, etc.) for retrieval tasks. It isn't necessary that what a word means globally also means the same in a local context. Building on the same line of work, relevance-based word embeddings [30] were explored to propose two new cost functions to learn word embedding specifically for retrieval tasks like query expansion and classification.

These days, mobile applications form a large chunk of data which is available for consumption. Some work has been done to mine user-reviews on mobile applications [27]. Mobile app retrieval has been experimented by few researchers to learn application representation using topic modeling [19] and intention in social media text [18]. Very recent work has been done to build a unified mobile search framework using neural approaches to learn high-dimensional representations for mobile apps [1]. Also, there are recent attempts made to use Learning-to-Rank algorithms for optimizing retrieval of mobile notifications triggered by installed apps [3]. However, application embeddings have been rarely utilized to perform query expansion, nearest neighbor analysis and other tasks related to mobile applications.

This paper proposes a novel method to learn dense word embeddings from app descriptions. The learned word embeddings are then used to compute relevance-based application embeddings which are suitable for retrieval and categorization of Mobile Apps.

### 3 Proposed Method

The following section details the method to extract word embeddings from app descriptions. As discussed in section 2 embedding techniques like word2vec and Glove are not suitable for information retrieval tasks as they are based on co-occurrence and fail to capture the associated relevance. It is observed that task specific embeddings generally perform better. To improve performance in tasks related to retrieval, it is important to learn these representations which carries the notion of relevance. With this motivation, the following subsection 3.1 discusses the process of learning this representation (word embedding) for each unique word in the vocabulary.

Once this sparse representation is calculated, it is used for learning a distributed vector representation using neural networks. This reduces noise, amplifies patterns and is suitable for mobile devices as computations involving dense vectors are faster and requires less space for storage. The approach proposed in this paper is language-independent as while formulating the

method it is assumed that application descriptions can be of any language and is seen as collection of words. Therefore, same techniques can be applied if datasets for different languages are available for training the models. For the methods discussed below, the terms - application embedding and document embedding are used in the same context.

### 3.1 Relevance-Based Word Representation

As it is said in linguistic theory [8]: “You shall know a word by the company it keeps”. For the domain of information retrieval, it can be said that a word is known by the documents (or other words) it retrieves. In order to capture the relevance associated with each word, the information contained in the top retrieved documents are utilized when the same word is used as a query to retrieval engine. Apache Lucene [2], an open source search library is used to test this hypothesis and get the relevant results for each word in the vocabulary. Lucene uses Okapi-BM25 score [21] to rank the retrieved documents. Using the app descriptions for all applications, Vector Space Model (VSM) representation is constructed using the term frequency [14] and inverse document frequency [25] (tf-idf) scores.

In this manner, every application is represented by a vector of dimension  $1 \times N$ , where  $N$  is the vocabulary size of the corpus. The weight vector for application  $j$  is defined as [24]:

$$VSM_j = [w_{1,j}, w_{2,j}, \dots, w_{N,j}], \quad (1)$$

$$w_{ij} = tf_{ij} \cdot \log \frac{TotalDocs}{|\{j' \in Doc \mid i \in j'\}|}, \quad (2)$$

where,  $tf(i, j)$  is the term frequency of term  $i$  in application  $j$ .  $TotalDocs$  is the total number of documents (applications) in the corpus.  $|\{j' \in Doc \mid i \in j'\}|$  is the total number of applications containing the term  $i$ .  $\log \frac{TotalDocs}{|\{j' \in Doc \mid i \in j'\}|}$  is inverse document frequency. For a query  $t$  with words  $t_1, t_2, \dots, t_{max}$ , BM25 score of document (application  $j$ ) is computed as:

$$BM25(t, j) = \sum_{i=1}^{max} idf(t_i) \cdot \frac{tf(t_i, j) \cdot (k_1 + 1)}{f(q_i, j) + k_1 \cdot (1 - b + b \cdot \frac{|j|}{avgdl})}, \quad (3)$$

$|j|$  is the length of document  $Doc$  in words,  $k_1$  and  $b$  are hyper-parameters and  $avgdl$  is the average document length.

#### 3.1.1 Method

Firstly, the most relevant documents ( $topDocs$ ) are retrieved after querying the system with each term in the vocabulary. The number of top documents is an input provided to the system and is given by the variable  $len(topDocs)$ . For the experiments in this paper  $len(topDocs)$  is set to 10. Using the constructed VSM and BM25 scores for these  $topDocs$ , a high-dimensional word representation is computed by the following equation:

$$WordRepr_k = \frac{\sum_{i=0}^{len(topDocs)} BM25_{i,k} \cdot VSM_i}{\sum_{i=0}^{len(topDocs)} BM25_{i,k}}, \quad (4)$$

where,  $VSM_i$  is the Vector Space Model representation for the  $i^{th}$  application and  $BM25_{i,k}$  is the BM25 score of the  $i^{th}$  document for term  $k$ . In this manner, word embeddings of all words in the vocabulary are computed. This word embedding takes into account the relevance of word in accordance with the context of data.

### 3.2 Dense Word Embedding

Curse of Dimensionality is a well known phenomenon in the field of Machine Learning and Data Mining. After a certain point, the increase in number of dimensions hurts the performance of algorithms. Inferring and performing computations on sparse vectors and matrices are costly operations as they may contain noise and irrelevant information. Even the vector given by equation 4 is sparse in nature which makes its usage limited. Hence, dimensionality reduction is used to remove noise and extract useful patterns. Different algorithms like Principal Component Analysis, Singular Value Decomposition, and Neural Networks are being used for performing the task of dimensionality reduction.

Firstly, to learn word embeddings from word representations, a simple yet effective feed-forward neural network architecture is used. The architecture is shown in Figure 1.

The input fed to the network is a  $N$  dimensional one hot encoded vector, where  $N$  is the vocabulary size and the output layer is  $N$  dimensional word representation calculated in equation 4. The activation functions used in the hidden and the output layer are ReLU and softmax respectively. Adam Optimizer has been used for updating the parameters of the neural network and the loss function used is cosine distance. The neural network tries to learn latent patterns present in the word representation while reducing dimension, making it computationally faster and cheaper for performing different retrieval tasks. For training the word embeddings, unique words from the app descriptions are used and passed as one hot encoded vector (input vector). The corresponding word representation is fed as the output of neural architecture. The dimension of the hidden layer is set to 300. Tensorflow framework [10] is used for the training process. For vocabulary size  $N$ , let the two weight matrices be  $W_1$  and  $W_2$  and  $t_j$  be the one-hot encoding of term  $j$  (input vector):

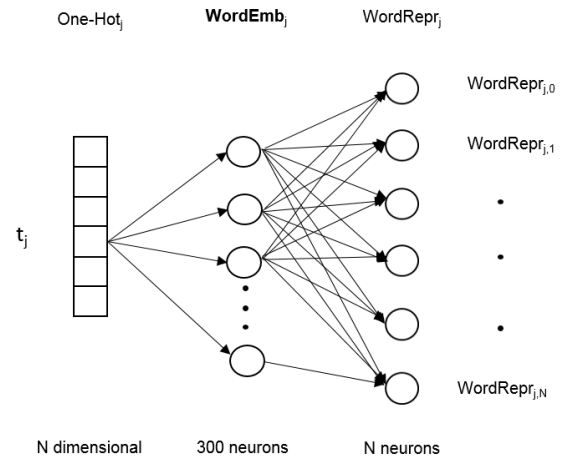
$$WordEmb_j = ReLu(t_j \times W_1), \quad (5)$$

where  $W_1$  has the dimension of  $N \times 300$  and  $ReLu(x) = \max(0, x)$  as given in [9]. The output layer is given by:

$$softmax(WordEmb_j \times W_2) + bias, \quad (6)$$

where  $W_2$  has the dimension of  $300 \times N$  and  $softmax$  is the activation function [5].  $bias$  is a vector of dimension  $1 \times N$ . The cosine distance loss is computed against the values from equation 6 and the representation stored in  $WordRepr_j$ . After training, the hidden layer gives the word embedding  $WordEmb$  for term  $j$ . This relevance based embedding has been developed keeping in mind the notion of relevance. This embedding is not intuitively built for machine learning tasks such as classification and clustering. To tackle problems such problems, a vanilla auto-encoder is trained to learn the word embeddings from the word representations proposed in equation 4. The input and output to this auto-encoder are the word representations given by equation 4. The embeddings learned in the hidden layer of the auto-encoder are denoted by  $WordEmb_{AE}$  for the rest of this paper.

With the discussed  $WordEmb_j$  representation for each term  $j$  in vocabulary, an embedding matrix is constructed.



**Fig. 1.** Neural architecture to generate Word Embedding  $WordEmb$  from Word Representation  $WordRepr$

### 3.3 RelEmb: Application Embedding

The intention behind any user query is to get the most relevant applications. Application title or category is not sufficient to extract relevant applications. But, each application has an associated description, which carries the most useful information about the application and its features. This description can be considered as bag of words and can be used to generate word embeddings as discussed in sections 3.1 and 3.2. The learned word embeddings can be extended for calculating application-level embedding by aggregating the word embedding of each term in the application's description. Let an application  $k$  (document) be represented as  $description(k) = \{w_1, w_2, w_3, \dots, w_i, \dots, w_n\}$  where each  $w_i$  is the word from vocabulary.

$$RelEmb_k = \frac{\sum_{i=0}^{len(description(k))} WordEmb[w_i]}{len(description(k))}. \quad (7)$$

Similarly, when the word embeddings are learned using vanilla auto-encoder, application embed-

dings are given by:

$$RelEmbAE_k = \frac{\sum_{i=0}^{len(description(k))} WordEmbAE[w_i]}{len(description(k))}. \quad (8)$$

## 4 Experimental Details

To evaluate the performance of the proposed method, extensive experiments are performed for different tasks related to application retrieval and categorization. A publicly available apps dataset is used for testing the performance of the embeddings. The dataset includes the data for query-application relevance judgment, which is useful to test the retrieval task of Query Expansion. A qualitative experiment in terms of nearest neighbor analysis is also elaborated in this paper. This shows the capability of app embedding for tasks like application recommendation. The dataset also includes a category tag for each application which can be used to analyze the performance of application embeddings for the task of multi-class app classification. Apart from supervised learning, these embedding vectors can also aid in unsupervised tasks like app clustering. The evaluation of embeddings has been discussed in this section. Results indicate superior performance as compared to existing state-of-the-art methods for various tasks.

### 4.1 Dataset

Data Set for Mobile App Retrieval [19] (TIMAN dataset) is used for evaluation of the proposed methods. This data include information about 43,041 mobile apps including the title, description, category, package name, user-reviews, query-document relevance pairs, etc. To trim down the vocabulary size, only English words from the app descriptions are selected. Moreover, minimum permissible length for each word was set to two. With the above mentioned preprocessing, the number of unique apps comes down to 42,895 with a vocabulary size of 37,213.

### 4.2 Application Retrieval

In mobile apps scenario, user generally queries with short-text with mostly 1-2 terms. It becomes a challenge for any retrieval engine to bring the most relevant results at the top ranks. Solution like query expansion helps to re-formulate the user query which eventually improves the retrieval accuracy. But, it is also problematic to bring the useful expanded terms which increases the relevance of the results. The word embeddings which are discussed in sections 3.1 and 3.2 are based on pseudo-relevance feedback (BM25) and are trained in a manner suitable for retrieval tasks. Authors believe that these embeddings are capable of finding the best expansion terms for user-query.

#### 4.2.1 Evaluation with Query Expansion

Query expansion is a standard retrieval task which has its practical advantages. It boosts the precision of a retrieval engine using a two-pass method. Query expansion approach is also an appropriate methodology to validate the performance of a retrieval system. Using the word embeddings proposed in section 3, query expansions tasks are evaluated. TIMAN dataset also provides the query-application relevance data with 4533 instances. The relevance score is a real-valued number in the range of 0-2. Since, it is a floating point (and not just binary) relevance label, *NDCG* metric [11] is used to judge the effectiveness of query expansion. The number of expanded terms is given by the variable  $k$ . To find the expansion terms for a given query, the following methodology is used:

- Calculation of input query vector using multiple terms in the user-query  $query = \{w_1, w_2, \dots\}$ .

$$Q = \frac{\sum_{i=0}^{len(query)} WordEmb[w_i]}{len(query)},$$

where,  $WordEmb[w_i]$  are calculated in equation 5.

- Given the query vector  $Q$ , the most semantically similar terms are found by calculating cosine similarity on the  $WordEmb$  matrix.

**Table 1.** Evaluation results on Query Expansion

Metric	Okapi-BM25	Expansion Techniques (k=5)			
		SVD	PCA	WordEmbAE	WordEmb
NDCG@3	0.569	0.499	0.467	0.572	<b>0.577</b>
NDCG@5	0.542	0.502	0.469	0.548	<b>0.563</b>
NDCG@10	0.535	0.505	0.477	0.542	<b>0.556</b>

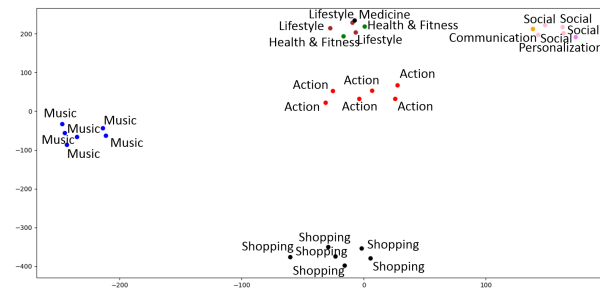
- Top  $k$  terms from the *WordEmb* matrix are selected based on their values of cosine similarity with respect to the input query vector  $Q$ .
- Example: *query* = {music, stream, airplay} retrieves top five expansion terms as {sonos, bose, dlna, player, listen}. It can be seen that these terms are closely related to the initial user query and all of them deal with music and streaming services.

Query expansion results are shown in Table 1. Okapi-BM25 scores are used as baseline by considering the input query  $Q$  as the direct query to the retrieval system (without expansion). The novelty of the proposed word embeddings can be seen from the increase in NDCG scores after query expansion.

*WordEmb* (equation 5) performed better than *WordEmbAE* as the intuition behind learning these embedding was the notion of relevance for the tasks in information retrieval. Query expansion is the right measure to validate this claim and the results fully support it. Even then, *WordEmbAE* was able to out-perform the baseline BM25 scores, proving the benefits of the sparse embeddings discussed in section 3.1.

#### 4.2.2 Nearest Neighbor Analysis

Not only quantitatively, but also qualitatively, the proposed application embeddings have proved its worth. It is not always the case that user provides a query term for app retrieval, a user may want to get some recommendations based on a particular app. Similar to [4], nearest neighbor analysis acts as recommender tool to get the closest match

**Fig. 2.** Visual representation for applications and their categories given in Table 4

from specific application. This recommendation system being retrieval based, *RelEmb* can be employed as application embedding. Qualitative results for nearest neighbor analysis using *RelEmb* (equation 7) are shown in the Table 4. From the results it can be seen that for an application of a particular category (query), the closest matched apps mostly belong to the same category. For these selected applications, *RelEmb* embeddings are plotted in a two-dimensional space using t-SNE visualization [15] (shown in Figure 2). The visualization also shows accurate grouping for different categories of applications. This section shows the effectiveness of application embedding for extracting similar apps.

#### 4.3 Application Categorization

Under Application categorization, classification and clustering are the two subdomains, which are most useful. Both classification and clustering use application embeddings as the feature vector for training.



**Table 2.** Evaluation results on App Classification (41 categories)

Classifier	F1 Score (Percentage)		
	Doc2Vec	RelEmb	RelEmbAE
Decision Tree	8.536	10.351	<b>21.332</b>
Multi-class SVM	32.550	40.557	<b>43.866</b>

**Table 3.** Evaluation results on Clustering Techniques

Metric	Different Methods		
	Doc2Vec	RelEmb	RelEmbAE
Silhouette Score	-0.0903	-0.061	<b>0.289</b>
Davies Bouldin Score	4.84	4.376	<b>0.8947</b>

Classification is a supervised learning algorithm which uses the document (or application) embedding as feature vector and the application category as the output variable. Trained application classifier can act as black box to predict the category of any new application. On the other hand, clustering is an unsupervised approach based on just the feature vectors. It can be used for grouping similar applications together, which has many practical use-cases like folder creation for grouping similar apps in mobile.

#### 4.3.1 App Classification

Decision tree and multi-class SVM are used to train the classifiers with features being application embedding and labels as application category. The considered TIMAN dataset has application data divided into 41 categories. To evaluate the performance of the proposed *RelEmb* and *RelEmbAE*, the dataset is divided into training and test splits using K-Fold cross validation ( $k = 5$ ). Multi-class classification using Doc2Vec embedding approach [13] is used as baseline for comparison. Evaluation results are presented in Table 2 and the reported F1-score is computed by averaging F1-score from all 5 groups (after K-Fold). For multi-class SVM classifier, the F1-score (in percentage) for *RelEmbAE* is 43.866 outperforming the baseline Doc2Vec, which gives the F1-score of 32.55. Similarly, for Decision tree the F1-score (in percentage) for *RelEmbAE* is

21.332 whereas for baseline Doc2Vec it is 8.536. The results indicate the better performance of *RelEmbAE* over Doc2Vec and also *RelEmb*.

#### 4.3.2 App Clustering

K-mean and DBSCAN clustering algorithms are used to evaluate the performance of application embedding for the task of clustering. The results with proposed *RelEmb* and *RelEmbAE* are compared with the baseline Doc2Vec embeddings. Two different metrics, such as Silhouette [23] and Davies Bouldin [6] scores are used to compare the results and the evaluation results on App Clustering are shown in Table 3. It is known that the value of silhouette score should be more closer to 1 for an accurate clustering. The scores for *RelEmbAE* are positive indicating better clustering in comparison to Doc2Vec for which the scores are negative, indicating wrongly assigned clusters. For Davies Bouldin score, a value closer to zero means a better separation between the clusters indicating superior performance of clustering. The numbers for both the clustering techniques indicate that *RelEmbAE* outperforms existing Doc2Vec embedding by significant margins.

As discussed in section 3.3, *RelEmbAE* is more suitable for categorization tasks whereas *RelEmb* performs well for retrieved-based tasks such as query expansion. For classification and clustering tasks, *RelEmb* still beats the state-of-the-art

**Table 4.** Qualitative results with Nearest Neighbors Analysis using *RelEmb*

Application	Predicted Nearest Applications - Name (Category)				
cops robbers jail break <b>(Action)</b>	survival hungry games (Action)	dead target zombie (Action)	wanted survival games (Action)	cube duty ghost blocks (Action)	orange block prison break (Action)
real piano <b>(Music)</b>	drum (Music)	real guitar (Music)	congas bongos (Music)	tabla (Music)	hip hop beatz (Music)
relax rain nature sounds <b>(Lifestyle)</b>	relax night nature sounds (Lifestyle)	relax forest nature sounds (Lifestyle)	white noise (Health & Fitness)	pain depression (Medical)	melodies sleep yoga (Health & Fitness)
love phrases images <b>(Social)</b>	top good night images (Social)	top good morning images (Social)	themes classic (Personal- ization)	status messages (Social)	video chat friendcaller (Communi- cation)
groger <b>(Shopping)</b>	ralphs (Shopping)	smith (Shopping)	king soopers (Shopping)	fred meyer (Shopping)	dillons (Shopping)

Doc2Vec approach showcasing the significance of initial relevance-based word representations learned in section 4 and denoted by *WordRepr*.

## 5 Conclusion and Future Work

In this paper, word embeddings are learned with a neural network architecture using the description of mobile applications. These embeddings are developed by keeping in mind the increasing usage of mobile applications and the difficulties faced to find out relevant ones from a large collection. The learning of word embeddings is carried out based on the notion of relevance. The results show that the learned word embeddings are effective for query expansion task eventually making the search experience more user friendly. In addition, the learned word embeddings are also aggregated to find RelEmbAE and RelEmb- distributed and dense representations of mobile application. These embeddings have outperformed Doc2vec on tasks like app classification and clustering.

In future, other parameters like application reviews, ratings, etc. can be used to make the application embedding much more rich and descriptive. Another extension of this work can be to analyze the use of application embeddings in the field of query intent detection, query classification which are current topics of research in Information Retrieval. Although the work in this paper is focused on tasks related to mobile applications, the same techniques can be applied to any generic scenarios of Information Retrieval.

## References

1. Aliannejadi, M., Zamani, H., Crestani, F., & Croft, W. B. (2018). Target apps selection: Towards a unified search framework for mobile devices. *The 41st International ACM SIGIR Conference on Research & Development in Information Retrieval*, SIGIR '18, ACM, New York, NY, USA, pp. 215–224.
2. Apache (2019). Lucene.

3. Bajaj, A., Tiwari, H., & Vala, V. (2019). Enhanced learning to rank using cluster-loss adjustment. *Proceedings of the ACM India Joint International Conference on Data Science and Management of Data*, ACM, pp. 70–77.
4. Biswas, A., Bhutani, M., & Sanyal, S. (2017). Mrnet-product2vec: A multi-task recurrent neural network for product embeddings. *Joint European Conference on Machine Learning and Knowledge Discovery in Databases*, Springer, pp. 153–165.
5. Bridle, J. S. (1990). Probabilistic interpretation of feedforward classification network outputs, with relationships to statistical pattern recognition. In *Neurocomputing*. Springer, pp. 227–236.
6. Davies, D. L. & Bouldin, D. W. (1979). A cluster separation measure. *IEEE transactions on pattern analysis and machine intelligence*, Vol. 2, pp. 224–227.
7. Diaz, F., Mitra, B., & Craswell, N. (2016). Query expansion with locally-trained word embeddings. *arXiv preprint arXiv:1605.07891*.
8. Firth, J. R. (1957). A synopsis of linguistic theory 1930-1955 in studies in linguistic analysis, philological society.
9. Glorot, X., Bordes, A., & Bengio, Y. (2011). Deep sparse rectifier neural networks. *Proceedings of the fourteenth international conference on artificial intelligence and statistics*, pp. 315–323.
10. Google (2019). Tensorflow. <https://www.tensorflow.org/>.
11. Järvelin, K. & Kekäläinen, J. (2002). Cumulated gain-based evaluation of ir techniques. *ACM Transactions on Information Systems (TOIS)*, Vol. 20, No. 4, pp. 422–446.
12. Kuzi, S., Shtok, A., & Kurland, O. (2016). Query expansion using word embeddings. *Proceedings of the 25th ACM international on conference on information and knowledge management*, ACM, pp. 1929–1932.
13. Le, Q. & Mikolov, T. (2014). Distributed representations of sentences and documents. *International Conference on Machine Learning*, pp. 1188–1196.
14. Luhn, H. P. (1957). A statistical approach to mechanized encoding and searching of literary information. *IBM Journal of research and development*, Vol. 1, No. 4, pp. 309–317.
15. Maaten, L. v. d. & Hinton, G. (2008). Visualizing data using t-sne. *Journal of machine learning research*, Vol. 9, No. Nov, pp. 2579–2605.
16. Mikolov, T., Sutskever, I., Chen, K., Corrado, G. S., & Dean, J. (2013). Distributed representations of words and phrases and their compositionality. *Advances in neural information processing systems*, pp. 3111–3119.
17. Mitra, B., Nalisnick, E., Craswell, N., & Caruana, R. (2016). A dual embedding space model for document ranking. *arXiv preprint arXiv:1602.01137*.
18. Park, D. H., Fang, Y., Liu, M., & Zhai, C. (2016). Mobile app retrieval for social media users via inference of implicit intent in social media text. *Proceedings of the 25th ACM International on Conference on Information and Knowledge Management*, ACM, pp. 959–968.
19. Park, D. H., Liu, M., Zhai, C., & Wang, H. (2015). Leveraging user reviews to improve accuracy for mobile app retrieval. *Proceedings of the 38th International ACM SIGIR Conference on Research and Development in Information Retrieval*, ACM, pp. 533–542.
20. Pennington, J., Socher, R., & Manning, C. (2014). Glove: Global vectors for word representation. *Proceedings of the 2014 conference on empirical methods in natural language processing (EMNLP)*, pp. 1532–1543.
21. Robertson, S. E., Walker, S., Jones, S., Hancock-Beaulieu, M. M., Gatford, M., et al. (1995). Okapi at trec-3. *Nist Special Publication Sp*, Vol. 109, pp. 109.
22. Rocchio, J. J. (1971). Relevance feedback in information retrieval. *The SMART retrieval system: experiments in automatic document processing*, pp. 313–323.
23. Rousseeuw, P. J. (1987). Silhouettes: a graphical aid to the interpretation and validation of cluster analysis. *Journal of computational and applied mathematics*, Vol. 20, pp. 53–65.
24. Salton, G., Wong, A., & Yang, C.-S. (1975). A vector space model for automatic indexing. *Communications of the ACM*, Vol. 18, No. 11, pp. 613–620.
25. Sparck Jones, K. (1972). A statistical interpretation of term specificity and its application in retrieval. *Journal of documentation*, Vol. 28, No. 1, pp. 11–21.
26. Statista (2018). Number of available applications in the Google Play Store from December 2009 to December 2018.
27. Vu, P. M., Nguyen, T. T., Pham, H. V., & Nguyen, T. T. (2015). Mining user opinions in mobile app reviews: A keyword-based approach (t). *2015 30th*

- IEEE/ACM International Conference on Automated Software Engineering (ASE)*, IEEE, pp. 749–759.
28. **Vulić, I. & Moens, M.-F. (2015).** Monolingual and cross-lingual information retrieval models based on (bilingual) word embeddings. *Proceedings of the 38th international ACM SIGIR conference on research and development in information retrieval*, ACM, pp. 363–372.
29. **Zamani, H. & Croft, W. B. (2016).** Embedding-based query language models. *Proceedings of the 2016 ACM international conference on the theory of information retrieval*, ACM, pp. 147–156.
30. **Zamani, H. & Croft, W. B. (2017).** Relevance-based word embedding. *Proceedings of the 40th International ACM SIGIR Conference on Research and Development in Information Retrieval*, ACM, pp. 505–514.
31. **Zheng, G. & Callan, J. (2015).** Learning to reweight terms with distributed representations. *Proceedings of the 38th international ACM SIGIR conference on research and development in information retrieval*, ACM, pp. 575–584.

Article received on 21/01/2019; accepted on 04/03/2019.  
Corresponding author is Shubham Krishna.

# Script Independent Morphological Segmentation for Arabic Maghrebi Dialects: An Application to Machine Translation

Salima Harrat<sup>1</sup>, Karima Meftouh<sup>2</sup>, Kamel Smaïli<sup>3</sup>

<sup>1</sup> École Normale de Bouzaréah, Algiers,  
Algeria

<sup>2</sup> Badji Mokhtar University-Annaba,  
Algeria

<sup>3</sup> Campus scientifique LORIA,  
France

slmhrrt@gmail.com, karima.meftouh@univ-annaba.dz, kamel.smaili@loria.fr

**Abstract.** This research deals with resources creation for under-resourced languages. We try to adapt existing resources for other resourced-languages to process less-resourced ones. We focus on Arabic dialects of the Maghreb, namely Algerian, Moroccan and Tunisian. We first adapt a well-known statistical word segmenter to segment Algerian dialect texts written in both Arabic and Latin scripts. We demonstrate that unsupervised morphological segmentation could be applied to Arabic dialects regardless of used script. Next, we use this kind of segmentation to improve statistical machine translation scores between the three Maghrebi dialects and French. We use a parallel multidialectal corpus that includes six Arabic dialects in addition to MSA and French. We achieved interesting results. Regards to word segmentation, the rate of correctly segmented words reached 70% for those written in Latin script and 79% for those written in Arabic script. For machine translation, the unsupervised morphological segmentation helped to decrease out-of-vocabulary words rates by a minimum of 35%.

**Keywords.** Arabic dialects, morphological segmentation, machine translation.

## 1 Introduction

The linguistic situation of the Arab world is characterized by the diglossia phenomenon, which is the co-existence of two variants of the same

language. A standard language (standard Arabic) used in formal speeches, newspapers, education, etc. Arabic dialects which are informal languages used in everyday conversations. Natural language processing of Arabic language does not take into account a large wide of these dialects, which lack NLP resources until today. These vernaculars are considered as under-resourced languages. Compared to other under-resourced languages, these dialects bring particular challenges because of their oral nature.

They were not written until the advance of Internet and mobile telephony. They have no standard rules that normalize their transcription therefore a word is written in different forms which are all acceptable.

Nowadays, Arabic dialects are widely used in social networks. They are written in Arabic and Latin script<sup>1</sup>. Also, they are written sometimes with a mixture of letters and numbers. Arab people exploit the similarity between some Arabic letters and numbers to write the dialect, for example similarity between 3 and ع, 7 and ح and 9 and ق. These dialects are variants of Arabic language;

<sup>1</sup>Arabic dialect written in Latin script is called in recent research: Arabizi, Arabish or Romanized Arabic.

they are different from it and also they differ from each other.

Maghrebi dialects are different from Middle-east dialects. Also, in the same Arab country several dialects exist. In addition, these dialects are evolving, new dialectal words appear every day and are adopted without academic validation. Even, Arabic dialects are influenced by other foreign languages such as French, Spanish, Turkish and Berber (for Maghrebi dialects). This influence generates the code-switching phenomenon, a dialectal sentence could include words from two or three languages. It is common to alternate between dialect, standard Arabic, French or English in the same conversation.

In this paper, we focus on Maghrebi Arabic dialects. We use a data-driven approach for word segmentation of Algerian dialect texts. We adapt Morfessor (a well-known statistical word segmenter); we present for the first time to our knowledge, a segmenter that considers dialectal texts written in both Arab and Latin scripts. In addition, we investigate the impact of word segmentation on machine translation performance. We use for this purpose statistical machine translation (SMT) from the three Maghrebi dialects (Algerian, Moroccan and Tunisian) to French.

To do this, we present a new version of a parallel corpus previously created and containing six Arabic dialects besides MSA. This new version includes for the first time a French text.

The rest of this article is organized as follows: we first describe briefly Arabic dialects, particularly, Magherbi ones (Section 2). Then, we provide an overview of related work on Arabic dialect morphological segmentation. Section 3 is dedicated to the adaptation of Morfessor for unsupervised and semi-supervised segmentation of Algerian dialect texts. In Section 4, we investigate the impact of morphological segmentation on statistical machine translation from Maghrebi dialects to French. Section 5 concludes this paper by pointing future directions of our work.

## 2 Arabic Dialects, Focus on the Three Main Maghrebi Dialects

Arabic dialects (vernaculars or colloquial Arabic) are considered as one of three variants of Arabic language, which includes also classical Arabic and Modern standard Arabic (MSA).<sup>2</sup> Arabic dialects are a spoken form of Arabic, used in everyday conversations, they are different from one Arabic country to another. They are influenced by both local tongues and foreign languages such as Spanish, French, Italian and English.

In terms of classification, Arabic dialects are distinguished regards to the East-west dichotomy[23]: (a) Middle-east dialects which include spoken Arabic of Arab Gulf countries and Yemen, Iraqi dialect, Levantine dialect (Syria, Lebanon, Palestine and Jordan), besides Egyptian and Sudanese dialects. (b) Maghrebi dialects which include the dialects of Algeria, Tunisia, Morocco, Libya and Mauritania.

As already mentioned before, we focus in this paper on the three main Maghrebi dialects: Algerian, Tunisian and Moroccan, one can raise the question: why only these dialects? The reason is that these dialects are the only ones for which we have relatively available resources.

In addition, the three Maghrebi countries share a lot of social, cultural, religious and linguistic similarities. Regarding the linguistic side, in the three countries, the Berber is the oldest language which has coexisted until now with the Arabic language bring to the region with Islamic conquest. The Algerian, Tunisian and Moroccan dialects are mutually intelligible, speakers of the three countries can readily understand each other. They share a lot of common features, even though they are different from each other. More extensive comparative details of the three dialects could be found in [20].

<sup>2</sup>Classical Arabic is the Arabic of the Quran and the ancient literature of Arabian peninsula while MSA is a modern form of classical Arabic.

### 3 Morphological Segmentation of Arabic Dialect Texts

#### 3.1 Related Work

Many efforts have been dedicated to build morphological segmenters for Arabic dialects texts. There are for this issue two main approaches; building segmenters from scratch [17, 5] or adapting MSA ones to take into account dialectal features.

Several studies adopted this last approach. Authors of [35] used the well known morphological analyzer BAMA[38] by extending its affixes tables to Levantine and Egyptian dialects. In the same way, BAMA was adapted to deal with Algerian dialect [19], the authors rebuilt affixes and stems tables. They kept MSA entries that apply also to Algerian dialect and integrated purely dialectal entries. Similarly, in [4], Al-Khalil morphological segmenter [8] has been adapted by enriching its affixes dictionary with a list of affixes belonging to four Arabic dialects. Likewise, the authors of the work described in [15] converted an Egyptian lexicon (ECAL, Egyptian Colloquial Arabic Lexicon) into a representation similar to the SAMA [14] dictionary (Standard Modern Arabic Analyzer). It should be noted that all these segmenters are dedicated to texts written in Arabic script.

#### 3.2 Motivation

Our goal is to segment Algerian dialects texts regardless of their script. To that end, we adopt an adaptive approach. However, we do not adapt a MSA morphological segmenter but rather a morphological segmenter based on probabilistic machine learning methods. The following reasons justify this choice:

- As mentioned above, dialectal texts are written in different forms with no standard orthography. They are written with Arabic and Latin script and sometimes with numbers instead of letters. This lack of writing rules is a challenging issue for morphological segmentation. Hence, data-driven approaches seem to be the most appropriate solution for this task.

- Non-standard spelling of dialects texts makes rule-based approach difficult to consider.
- Because of the evolving nature of dialectal vocabulary, new words appear and are rapidly spread in speakers' community. Data-driven methods could easily take into account this words and their inflected forms.

#### 3.3 Morfessor

In this respect, we opted for Morfessor, a well-known morphological segmenter suitable for languages with complex morphology like Finnish and Turkish. It has been integrated into different NLP applications like speech recognition [24, 30, 13, 36], machine translation [41, 27, 29, 9, 33] and speech retrieval [7, 39].

Morfessor [10, 11] is a set of statistical methods for segmenting words based on the Minimum Description Length principle. It learns morphemes from data in an unsupervised manner. The level of segmentation is tuned by adjusting the weight  $\alpha$  between the cost of encoding the lexicon (the parameters  $\Theta$ ) and the cost of encoding the training data ( $D$ ) part in the cost function:

$$L(\Theta, D_w) = -\log P(\Theta) - \alpha \log P(D_w|\Theta). \quad (1)$$

An interesting version of Morfessor is that described in [26], where a semi-supervised training approach is used. The above function cost is summarized as follows:

$$L(\Theta, D_w) = -\log P(\Theta) - \alpha \log P(D_w|\Theta) - \beta \log P(A|\Theta), \quad (2)$$

where  $A$  is the annotated training data, and  $\alpha$  and  $\beta$  in this order, are the weights of the unannotated and annotated data training. In the context of this work, we use the Morfessor 2.0 implementation [40].

#### 3.4 Data Description

In order to train Morfessor, we used textual corpora recently created in the context of processing Algerian dialect. Below, we give an overview of each corpus.

- The comparable corpus CALYOU  
CALYOU<sup>3</sup>[1] is an Algerian dialect comparable

<sup>3</sup>Comparable spoken ALgerian extracted from YOUTube

corpus of Youtube comments. It was collected by querying Youtube with key-words related to current Algerian events. The corpus includes comments written with Arabic script aligned to ones written with Latin script. This alignment is got by using word embeddings. We give in Tables 1 and 2 respectively, statistics about this corpus and some comments examples written with Arabic and Latin scripts including even numbers.

**Table 1.** CALYOU corpus statistics

#Comments(K)	#Words(M)	#Distinct words(K)
853	12.7	88

**Table 2.** Examples of CALYOU comments with mapping between Arabic letters and numbers

Dialectal comment	Meaning
فور بزاف	It is very good
خاوتي ديروا ابوني	Brothers! subscribe
علاخطش نحبو بلادنا بزافو	Because we love our country
بزافو	bravo
3andk l7a9 (3=ع , 9=ق)	You are right
ya3tik asaha madame (3=ع)	Thank you madam
sa7bi a9ra l'histoire ta3 bladek (7=ح)	My friend, you must learn the history of your country
taktal badahk 5ouya (5=خ)	You are very funny my brother

— Algerian text of PADIC (ALG-PADIC) PADIC<sup>4</sup>[28] is multidialectal Arabic corpus including Algerian, Tunisian, Moroccan, Syrian and Palestinian in addition to MSA. We use for the purpose of this work, the Algerian side of this corpus (see statistics in Table 3).

### 3.5 Experimentation

The experiments were carried out using the corpora described above. We experimented

<sup>4</sup>Parallel Arabic Dialect Corpus, downloadable on <http://smart.loria.fr/pmwiki/pmwiki.php/PmWiki/>

**Table 3.** ALG-PADIC corpus statistics

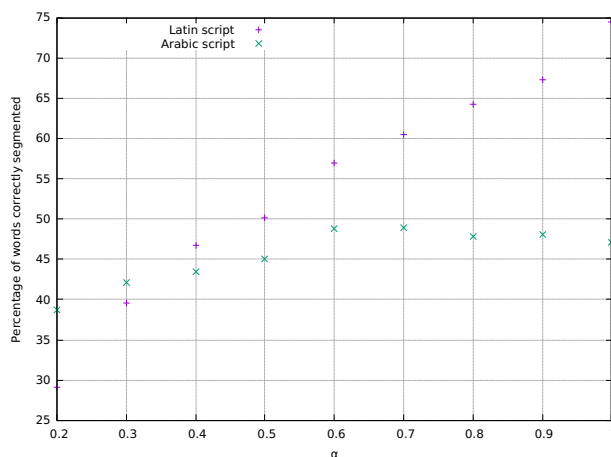
#Sentences(K)	#Words(K)	#Distinct words(K)
6.4	40.75	9.15

unsupervised segmentation trained with CALYOU corpus. Then, we conducted a semi-supervised training of Morfessor with annotated data provided from the ALG-PADIC corpus.

For evaluation purpose, we randomly extracted two datasets of 200 CALYOU comments written in Arabic and Latin scripts with respectively, 1730 and 1609 words. The two test datasets has been segmented by hand.

#### 3.5.1 Unsupervised Morphological Segmentation

We trained Morfessor with CALYOU corpus. In order to tune the weight  $\alpha$  that controls segments lengths (a low  $\alpha$  favors small construction lexicons, while a high value favors longer constructions), we made several experiments starting with the default value ( $\alpha = 1$ ). The figure 1 retraces the results in terms of percentage of correctly segmented words written in Latin and Arabic scripts according to the different values of  $\alpha$ .



**Fig. 1.** Percentages of correctly segmented words using unsupervised method with different values of  $\alpha$

It shows that 74.46% of words written with Latin script in the test set are correctly segmented for the



default value of  $\alpha$  which proved to be the best of all the ones we tested.

Indeed, the segmentation takes into account several morphological features like function words inflection. We give in Table 4 some examples of valid segmentations provided by the test set.

Furthermore, for invalid segmentations, we noticed that in most cases Morfessor could identify some segments of the word even though he could not identify all the segments. For example, the circumfix negation affixes are often distinguished (see examples in Table 5).

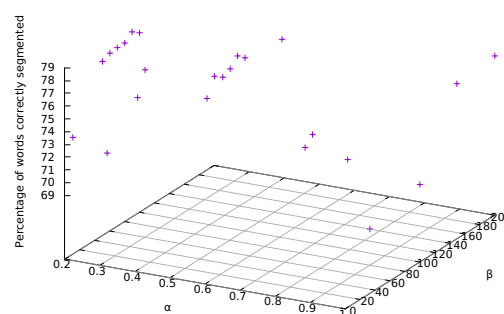
For Words written with Arabic script, unsupervised segmentation performs worse than words with Latin script. The best-recorded percentages are got for an  $\alpha$  value of 0.7 and do not reach 50%. However, even for Arabic script words, Morfessor could identify correctly some segments of a word although the whole segmentation is not valid. Tables 6 and 7 show some illustrative examples.

### 3.5.2 Semi-supervised Morphological Segmentation

In this experiment, we performed tests with semi-supervised segmentation. Unfortunately, we did it for dialect texts written with Arabic script only, since annotation data for Latin script are not available for us. Indeed, we used the ALG-padic corpus for annotation. We have segmented it using the morphological analyzer [19] described earlier<sup>5</sup>.

Morfessor is thus trained with CALYOU corpus and ALG-padic annotated corpus. It should be noted that in addition to the  $\alpha$  parameter already described, Morfessor uses another parameter  $\beta$  that controls the contribution of the annotation data in the segmentation operation. We first started by using the segmentation with the default values, then we experimented different values of  $\alpha$  and  $\beta$ . We show in Figure 2 the best achieved results in terms of percentages of correctly segmented words.

Semi-supervised segmentation shows promising results regards to the size of the annotated corpus. Indeed, the best percentage of correctly segmented word reached 78.55%. According to



**Fig. 2.** Percentages of correctly segmented words using semi-supervised method with different values of  $\alpha$  and  $\beta$

the test sample, Semi-supervised segmentation could take into account many dialectal morphological features. Most agglutinated forms that the unsupervised segmentation failed to segment where correctly analyzed by the semi-supervised analysis. We report in Table 8 some examples.

Furthermore, despite its ability to segment agglutinated forms correctly, we noticed that even with semi-supervised analysis, the negation forms is difficult to segment for words written in Arabic script. Morfessor failed to identify all word segments. Some examples are reported in Table 9.

We also found that for some words, semi-supervised analysis tends to over-segment. In Table 10 are given some examples of these cases.

Morfessor segmentation seems to be an interesting direction for segmenting dialectal Arabic words, in view of the fact that it can be used for texts in Arabic and Latin scripts. Moreover, transcribing dialect by introducing numbers is not problematic with Morfessor. Words written with numbers are segmented as well as words including only letters. Verb conjugation and noun declension are taken into account as illustrated above in the various examples. In addition, through the different segmentations that we analyzed, the agglutinative forms of the dialects (more complicated than MSA) are for the most part parsed.

<sup>5</sup>We remind that this Analyzer supports the Arabic script only.

**Table 4.** Examples of words written with Latin script correctly segmented using unsupervised method

Segments	Word	Segmentation	Meaning
Conjunction+demonstrative pronoun.	whada	w+hada	And this one
Conjunction+noun	wrajel	w+rajel	And a man
Definition article+noun	l3sal	l+ 3sal	The honey
Function word+pronoun	3andna	3and+na	We have
Preposition+pronoun	mnhoum	mn+houm	From them
Subject-prefix+verb	yadkhol	ya+dkhol	He enters
Verb+suffix-subject	kbarty	kbar+ty	You have grown

**Table 5.** Examples of words written with Latin script partially segmented using unsupervised method

Word	Partial segmentation	Valid segmentation	Meaning
mayjouzch	ma+yjouz+ch	ma+y+jouz+ch	He does not pass
yaatik	ya+atik	ya+ati+k	He gives you
may3arfakch	may+3arfak+ch	ma+y+3arf+ak+ch	He does not know you

**Table 6.** Examples of words written with Arabic script correctly segmented using unsupervised method

Segments	Word	Segmentation	Meaning
Conjunction + personal pronoun	وانت	و+انت	And you
Definition article+noun	الدار	ال+دار	The house
Function word+pronoun	عندي	عند+ي	I have
Verb+subject suffix+ object suffix	سبقونا	سبق+و+نا	They have surpassed us
Preposition+noun	بذراهم	ب+دراهم	With money
Preposition+noun+ suffix pronoun	لبنتك	ل+بنت+ك	For your daughter

**Table 7.** Examples of words written with Arabic script partially segmented using unsupervised method

Word	Partial segmentation	Valid segmentation	Meaning
تعيشي	ت+عيشي	ت+عيش+ي	You live
جربتو	جرب+تو	جرب+ت+و	I tried it
وماجاش	و+ماجا+ش	و+ما+جا+ش	And he did not come

#### 4 Impact of Morphological Segmentation on SMT of Maghrebi Dialects to French

Word segmentation is an important step in many NLP tasks related to Arabic. Many work show that it improves performance of NLP applications like part-of-speech tagging [12, 16, 34] and machine translation [18, 2, 3, 34].

In this respect, we attempt to measure the impact of unsupervised segmentation on machine translation performance in the context of translating between Arabic Maghrebi dialects (Algerian, Tunisian and Moroccan) and French.

It should be noted that, most research efforts in this area concern English. For more details, the reader is referred to [21] where a comprehensive survey on Arabic dialects machine translation is presented.

**Table 8.** Examples of words correctly segmented with semi-supervised method

Segments	Word	Segmentation	Meaning
Conjunction+noun+plu. suffix+pronoun suff.	ووليدآتك	و+وليد+آت+ك	And your children
Conj.+subj.pref.+verb+subj. suff.+obj. suff.	ونعآودوها	و+ن+عآود+وها	And we will repeat it
Subj. pref.+ verb+ object suff.	يخلصهم	ي+خلص+هم	He pays them
Function word+pronoun suff.	علينآ	علي+نآ	On us
Def. article+noun+plural suff.	المومنين	ال+مومن+ين	The believers

**Table 9.** Examples of words written with Arabic script partially segmented with semi-supervised method

Word	Partial segmentation	Valid segmentation	Meaning
وتتمآلك	و+ن+تمن+ال+ك	و+ن+تمنآ+ل+ك	I wish you
مآتحكيليش	مآت+حكي+ل+ي+ش	مآت+حكي+ل+ي+ش	Do not tell me
مآتحيوش	مآت+حبو+ش	مآت+ح+ب+و+ش	You do not like it

**Table 10.** Example of over-segmented words with semi-supervised method

Word	Over Segment.	Valid Segment.	Meaning
سخون	سخ+و+ن	سخون	Hot
معسله	مع+سل+ه	معسل+ه	Honeydew
برآفو	ب+رآف+و	برآفو	Bravo

#### 4.1 Settings

We use a phrase-based statistical machine translation [25], with Giza++[31] for alignment and KenLM [22] to compute ngram language models. We also use an unsupervised segmentation with Morfessor. We choose unsupervised segmentation because annotation data are not available for Tunisian and Moroccan dialects, they are available only for Algerian dialect.

#### 4.2 Data Description

1. Parallel corpus of Maghrebi dialects and French:

We use the three Maghrebi dialect texts of PADIC corpus (Algerian, Moroccan and Tunisian) and for the first time, a parallel

French text translated from the standard Arabic side of PADIC (see Table 12).

2. Monolingual corpora:

For unsupervised training of Morfessor, three monolingual dialectal corpora are used. A brief description of these corpora is given below (with some statistics in Table 13):

- Arabic script part of CALYOU used earlier.
- A Tunisian corpus of facebook comments [6] collected during the period of Arab spring events (we used only comments written with Arabic script).
- A Moroccan corpus of texts [37] collected from different sources (web sites, plays and records of everyday conversations).

Also we used a French monolingual corpus which we downloaded from OPUS<sup>6</sup> web site to train French language models.

#### 4.3 Experimentation

We trained all the machine translation systems on 5.9K parallel sentences. We allocated 0.1K

<sup>6</sup><http://opus.nlpl.eu/>

**Table 11.** BLEU scores and OOV rates of Maghrebi dialects to French SMT according to different training data of Morfessor

SMT systems	Algerian		Moroccan		Tunisian	
	BLEU	OOV%	BLEU	OOV%	BLEU	OOV%
SMT-no-segmentation	6.90	24.7	9.01	23.5	7.43	28.3
SMT+seg(32K)	6.29	16.2	8.08	15.1	8.68	14.5
SMT+seg(62K)	7.31	12.6	8.84	15.5	-	-

**Table 12.** The parallel corpus statistics

Corpus	#Words (K)	#Distinct word (K)
Algerian	40.75	9.15
Moroccan	42.58	9.70
Tunisian	38.96	10.04
French	62.02	7.91

**Table 13.** Statistics of dialectal monolingual corpora used for unsupervised segmentation

Corpus	#Words(K)	#Distinct words (K)
Algerian	412.93	191.17
Moroccan	349.30	62.87
Tunisian	131.46	32.25

and 0.4K sentences for tuning and evaluation, respectively. The baseline SMT systems (SMT-no-segmentation) are trained on unsegmented data for the three dialects (source language being the dialect and the target language French). Next, we segmented data for training, tuning and evaluating SMT systems.

Regards to the size of monolingual dialectal corpora used for learning Morfessor, we conducted two types of experiments. In the first one, data of the three SMT systems (SMT+seg(32K)) are segmented by learning Morfessor with datasets of 32K distinct words for each dialect (32K is the size of Tunisian corpus, the smallest monolingual corpus).

In the second experiment, SMT systems (SMT+seg(62K)) data were segmented by learning Morfessor with datasets of 62K distinct words. This experiment concerns only Algerian and Moroccan dialects because we have no more data for Tunisian dialect. We evaluated all SMT systems described in terms of BLEU [32] metric.

Table 11 shows results. We notice that for Tunisian dialect-to-French translation, SMT system that uses Morfessor segmentation outperforms the system that does not use segmentation by 1.25 BLEU points. For Algerian-to-French and Moroccan-to-French SMT systems whose data were segmented by Morfessor learned with datasets of 32K words, BLEU scores decrease by 0.61 and 0.93 points respectively.

However, when Morfessor is learned with more dialectal data (62K words), BLEU score of Algerian-to-French increases by 0.41 compared to the baseline system score. For Moroccan-to-French translation, the BLEU score of SMT+seg(62K) system (segmentation learned on a dataset of 62K words) outperforms the SMT+seg(32K) system (segmentation learned on a dataset of 32K words).

But, the baseline system remains the best. Furthermore, Morfessor segmentation decreases significantly OOV rates. Indeed, OOV rates of Algerian-to-French and Tunisian-to-French SMT systems trained on segmented data decrease by almost 50%. For Moroccan-to-French SMT system, Morfessor does not improve BLEU scores as seen, but it decreases the OOV rates by at least 34%.

## 5 Conclusion

We have adopted an unsupervised and semi-supervised approach to segment Algerian dialect texts written in Arabic and Latin scripts. This work was accomplished by using Morfessor. The results are encouraging.

Indeed, most morphological features of Algerian dialects are taken into account. The semi-supervised segmentation applied only for text

written in Arabic script achieved the best results. We further evaluated the impact of unsupervised segmentation of Maghrebi dialect texts on SMT systems that translate from the three Maghrebi dialects to French. For the first time, we introduced a French text to PADIC corpus. This text was used to train the target side of the SMT systems. The unsupervised segmentation has improved BLEU scores especially for Tunisian-to-French and Algerian-to-French SMT systems. Moreover, the OOV rates decrease by nearly 50% for these two SMT systems and by more than 34% for the Moroccan-to-French SMT system. In the future, we would like to use an iterative process to create annotation data for Algerian dialect texts in Latin script. We will use unsupervised segmentation to segment the words, then valid outputs will be used to create annotation data. This will allow us to consider semi-supervised segmentation for these texts. In the same way, we will enrich the annotated data in Arabic script. Finally, the new version of PADIC will be made available to the scientific community.

## References

1. **Abidi, K., Menacer, M.-A., & Smaili, K. (2017).** Calyou: A comparable spoken Algerian corpus harvested from youtube. *18th Annual Conference of the International Communication Association (Interspeech)*.
2. **Al-Mannai, K., Sajjad, H., Khader, A., Al Obaidli, F., Nakov, P., & Vogel, S. (2014).** Unsupervised word segmentation improves dialectal Arabic to English machine translation. *Proceedings of the EMNLP 2014 Workshop on Arabic Natural Language Processing (ANLP)*, pp. 207–216.
3. **Almahairi, A., Cho, K., Habash, N., & Courville, A. (2016).** First result on Arabic neural machine translation. *arXiv preprint arXiv:1606.02680*.
4. **Almeman, K. & Lee, M. (2012).** Towards developing a multi-dialect morphological analyser for Arabic. *4th International Conference on Arabic Language Processing*, pp. 19–25.
5. **Altantawy, M., Habash, N., & Rambow, O. (2011).** Fast yet rich morphological analysis. *Proceedings of the 9th International Workshop on Finite State Methods and Natural Language Processing*, Association for Computational Linguistics, pp. 116–124.
6. **Ameur, H. & Jamoussi, S. (2013).** Dynamic construction of dictionaries for sentiment classification. *2013 IEEE 13th International Conference on Data Mining Workshops*, IEEE, pp. 896–903.
7. **Arisoy, E., Can, D., Parlak, S., Sak, H., & Saraclar, M. (2009).** Turkish broadcast news transcription and retrieval. *Trans. Audio, Speech and Lang. Proc.*, Vol. 17, No. 5, pp. 874–883.
8. **Boudlal, A., Lakhouaja, A., Mazroui, A., Meziane, A., Bebah, M. O. A. O., & Shoul, M. (2011).** Alkhalil morpho sys: A morphosyntactic analysis system for Arabic texts. *Proceedings of 7th International Computing Conference in Arab ACIT*.
9. **Clifton, A. & Sarkar, A. (2011).** Combining morpheme-based machine translation with post-processing morpheme prediction. *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies-Volume 1*, Association for Computational Linguistics, pp. 32–42.
10. **Creutz, M. & Lagus, K. (2002).** Unsupervised discovery of morphemes. *Proceedings of the ACL-02 workshop on Morphological and phonological learning-Volume 6*, Association for Computational Linguistics, pp. 21–30.
11. **Creutz, M. & Lagus, K. (2007).** Unsupervised models for morpheme segmentation and morphology learning. *ACM Transactions on Speech and Language Processing (TSLP)*, Vol. 4, No. 1, pp. 3.
12. **Diab, M., Hacioglu, K., & Jurafsky, D. (2004).** Automatic tagging of Arabic text: From raw text to base phrase chunks. *Proceedings of HLT-NAACL 2004: Short papers*, Association for Computational Linguistics, pp. 149–152.
13. **Gelas, H., Besacier, L., & Pellegrino, F. c. (2012).** Developments of Swahili resources for an automatic speech recognition system. *Spoken Language Technologies for Under-Resourced Languages*.
14. **Graff, D., Maamouri, M., Bouziri, B., Krouna, S., Kulick, S., & Buckwalter, T. (2009).** Standard Arabic morphological analyzer (SAMA) version 3.1. *Linguistic Data Consortium LDC2009E73*.
15. **Habash, N., Eskander, R., & Hawwari, A. (2012).** Morphological analyzer for Egyptian Arabic.

- Proceedings of the Twelfth Meeting of the Special Interest Group on Computational Morphology and Phonology SIGMORPHON*, Association for Computational Linguistics, pp. 1–9.
16. **Habash, N. & Rambow, O. (2005).** Arabic tokenization, part-of-speech tagging and morphological disambiguation in one fell swoop. *Proceedings of the 43rd Annual Meeting on Association for Computational Linguistics*, Association for Computational Linguistics, pp. 573–580.
  17. **Habash, N. & Rambow, O. (2006).** Magead: A morphological analyzer and generator for the Arabic dialects. *Proceedings of the 21st International Conference on Computational Linguistics and the 44th annual meeting of the Association for Computational Linguistics*, Association for Computational Linguistics, pp. 681–688.
  18. **Habash, N. & Sadat, F. (2006).** Arabic preprocessing schemes for statistical machine translation. *Proceedings of the Human Language Technology Conference of the NAACL, Companion Volume: Short Papers*, Association for Computational Linguistics, pp. 49–52.
  19. **Harrat, S., Meftouh, K., Abbas, M., & Smaili, K. (2014).** Building resources for Algerian Arabic dialects. *Proceedings of Interspeech*, pp. 2123–2127.
  20. **Harrat, S., Meftouh, K., & Smaili, K. (2018).** Maghrebi Arabic dialect processing: an overview. *Journal of International Science and General Applications*, Vol. 1.
  21. **Harrat, S., Meftouh, K., & Smaili, K. (2019).** Machine translation for Arabic dialects (survey). *Information Processing & Management*, Vol. 56, No. 2, pp. 262 – 273. Advance Arabic Natural Language Processing (ANLP) and its Applications.
  22. **Heafield, K. (2011).** Kenlm: Faster and smaller language model queries. *Proceedings of the Sixth Workshop on Statistical Machine Translation*, Association for Computational Linguistics, pp. 187–197.
  23. **Hetzron, R. (1997).** *The Semitic Languages*. Routledge language family descriptions. Routledge.
  24. **Hirsimäki, T., Creutz, M., Siivola, V., Kurimo, M., Virpioja, S., & Pytkönen, J. (2006).** Unlimited vocabulary speech recognition with morph language models applied to Finnish. *Computer Speech & Language*, Vol. 20, No. 4, pp. 515–541.
  25. **Koehn, P., Hoang, H., Birch, A., Callison-Burch, C., Federico, M., Bertoldi, N., Cowan, B., Shen, W., Moran, C., Zens, R., Dyer, C., Bojar, O., Constantin, A., & Herbst, E. (2007).** Moses: Open Source Toolkit for Statistical Machine Translation. *Proceedings of the Annual Meeting of the Association for Computational Linguistics, demonstration session*, pp. 177–180.
  26. **Kohonen, O., Virpioja, S., & Lagus, K. (2010).** Semi-supervised learning of concatenative morphology. *Proceedings of the 11th Meeting of the ACL Special Interest Group on Computational Morphology and Phonology*, Association for Computational Linguistics, pp. 78–86.
  27. **Luong, M.-T., Nakov, P., & Kan, M.-Y. (2010).** A hybrid morpheme-word representation for machine translation of morphologically rich languages. *Proceedings of the 2010 Conference on Empirical Methods in Natural Language Processing*, Association for Computational Linguistics, pp. 148–157.
  28. **Meftouh, K., Harrat, S., Jamoussi, S., Abbas, M., & Smaili, K. (2015).** Machine translation experiments on PADIC: A Parallel Arabic Dialect Corpus. *Proceedings PaCLiC 29th Asia Conference on Language, Information and Computation*, pp. 26–34.
  29. **Mermer, C. (2010).** Unsupervised search for the optimal segmentation for statistical machine translation. *Proceedings of the ACL 2010 Student Research Workshop*, Association for Computational Linguistics, pp. 31–36.
  30. **Mihajlik, P., Tuske, Z., Tarjan, B., Nemeth, B., & Fegyö, T. (2010).** Improved recognition of spontaneous Hungarian speech; morphological and acoustic modeling techniques for a less resourced task. *IEEE Transactions on Audio, Speech, and Language Processing*, Vol. 18, No. 6, pp. 1588–1600.
  31. **Och, F. J. & Ney, H. (2003).** A Systematic Comparison of Various Statistical Alignment Models. *Computational Linguistics*, volume 29, number 1, pp. 19–51.
  32. **Papineni, K. & al. (2001).** Bleu: a method for automatic evaluation of machine translation. *Proceedings of the 40th Annual of the Association for Computational linguistics*, Philadelphia, USA, pp. 311–318.
  33. **Popović, M. (2011).** Morphemes and POS tags for n-gram based evaluation metrics. *Proceedings of*

the Sixth Workshop on Statistical Machine Translation, Association for Computational Linguistics, pp. 104–107.

34. Sajjad, H., Dalvi, F., Durrani, N., Abdelali, A., Belinkov, Y., & Vogel, S. (2017). Challenging language-dependent segmentation for Arabic: An application to machine translation and part-of-speech tagging. *arXiv preprint arXiv:1709.00616*.
35. Salloum, W. & Habash, N. (2011). Dialectal to standard Arabic paraphrasing to improve Arabic-English statistical machine translation. *Proceedings of the First Workshop on Algorithms and Resources for Modelling of Dialects and Language Varieties*, Association for Computational Linguistics, pp. 10–21.
36. Smit, P., Leinonen, J., Jokinen, K., Kurimo, M., et al. (2016). Automatic speech recognition for Northern Sami with comparison to other Uralic languages. *Proceedings of the Second International Workshop on Computational Linguistics for Uralic Languages*, The Research Group on Artificial Intelligence (RGAI).
37. Tachicart, R., Bouzoubaa, K., Aouragh, S. L., & Jaafa, H. (2018). Automatic identification of Moroccan colloquial Arabic. *Arabic Language Processing: From Theory to Practice*, Springer International Publishing, pp. 201–214.
38. Tim, B. (2002). Buckwalter Arabic morphological analyzer version 1.0. *Linguistic Data Consortium LDC2002L49*.
39. Turunen, V. T. & Kurimo, M. (2008). Speech retrieval from unsegmented Finnish audio using statistical morpheme-like units for segmentation, recognition, and retrieval. *ACM Trans. Speech Lang. Process.*, Vol. 8, No. 1, pp. 1:1–1:25.
40. Virpioja, S., Smit, P., Grönroos, S.-A., Kurimo, M., et al. (2013). Morfessor 2.0: Python implementation and extensions for Morfessor Baseline.
41. Virpioja, S., Väyrynen, J. J., Creutz, M., & Sadeniemi, M. (2007). Morphology-aware statistical machine translation based on morphs induced in an unsupervised manner. *Machine Translation Summit XI*, Vol. 2007, pp. 491–498.

Article received on 24/01/2019; accepted on 04/03/2019.  
Corresponding author is Salima Harrat.





# Sentence Generation Using Selective Text Prediction

Samarth Navali, Jyothirmayi Kolachalam, Vanraj Vala

Samsung R&D Insitute, Bengaluru,  
India

{s.pnavali, jyothi.kolac, vanraj.vala}@samsung.com

**Abstract.** Text generation based on comprehensive datasets has been a well-known problem from several years. The biggest challenge is in creating a readable and coherent personalized text for specific user. Deep learning models have had huge success in the different text generation tasks such as script creation, translation, caption generation etc. Most of the existing methods require large amounts of data to perform simple sentence generation that may be used to greet the user or to give a unique reply. This research presents a novel and efficient method to generate sentences using a combination of Context Free Grammars and Hidden Markov Models. We have evaluated using two different methods, the first one is using a score similar to the BLEU score. The proposed implementation achieved 83% precision on the tweets dataset. The second method of evaluation being a subjective evaluation for the generated messages which is observed to be better than other methods.

**Keywords.** Text generation, sentence generation, context free grammar, CFG, hidden Markov model, HMM, selective text prediction.

## 1 Introduction

Machine Learning has found a lot of application in almost all the major industries such as finance, IT, healthcare, etc. Machine Learning has started to replace the traditional methods mainly because of the power of these algorithms. They are used for a variety of tasks such as generation, classification, localisation etc. They learn features from the training data and then use what it has learnt to perform the respective task. They were not preferred initially as there wasn't enough

computational power to run these algorithms. With computational power easily available these days, machine learning has emerged as the go-to solution.

Natural Language Processing(NLP) is one such application. In this application machine learning models try to gain some understanding from textual,voice data and then use it for various other applications. Machine learning for NLP may be used to generate text, identify parts of speech, to perform Named Entity Recognition(NER) etc. These kinds of tasks are used to gain intelligence from the text, voice based data and then utilise this information as and when required.

One of the tasks that come under NLP is generation of text. This generation is based on some existing textual data. At times it so happens that somebody wants to caption an image, but that person doesn't know what would be a good caption or when somebody wants to write a poem which seems as if it's written by Robert Frost. All these are tasks that involve generation of text. With the help of Machine Learning one can use the previous occurrences of the task and learn from it and generate text.

Neural Networks have been found to attain state-of-the-art results in a majority of the natural language processing tasks such as sentiment analysis, text generation etc. Within NLP, quite a number of tasks involve generating text based on some input data.

Neural Networks are really powerful and can gain more insights from the input data as compared to the other machine learning algorithms.

In the recent years Recurrent Neural Networks(RNN) and Long Short Term Memory Networks (LSTM) have taken over in such tasks. They have the power to remember the contexts from before and use that context details in generating text. Text is generally generated from these models that takes a sample from a distribution that is conditioned on the previous words and the hidden state consists of some representation of all the words generated so far. At times they are trained with a method called as teacher forcing, where the ground-truth words are fed back into the model for the generation of text. This causes problems as the model is forced to condition on sequences that were not initially observed during the training time. This leads to unpredictable outcomes in the hidden state of the RNN. Also as they require large amounts of data to train and generate presentable sentences, they were not preferred because of the availability of less data.

As the target for this research work is to generate simple sentences, it's not worth going through all the disadvantages of the RNN. Also as for such tasks we are not utilising the true power of RNN's and LSTM's, so using them does not make sense. So for these cases this research work describes a method where we combine Context Free Grammar's and Hidden Markov Models to generate sentences.

A **Context Free Grammar (CFG)** is a set of recursive rewriting rules which are used to generate a variety of strings. It is a quadruple  $(N, T, P, S)$  where:

- $N$  is a set of non-terminal symbols.
- $T$  is a set of terminals where  $N \cap T = \text{NULL}$ .
- $P$  is a set of rules,  $P: N \rightarrow (N \cup T)$ , i.e., the left-hand side of the production rule  $P$  does not have any right context or left context.
- $S$  is the start symbol.

We first start off with the starting symbol and then replace each of the starting symbol with the rule associated with that symbol.

Then proceed until no more substitutions can be made or until the desired terminal state is reached. CFG's are used to generate patterns of strings, pattern matching etc.

**Hidden Markov Model (HMM)** is a method for representing probability distributions over a sequences of observations. It gets its name from two main properties. The first one being, the prediction for the current state comes from a state  $S_t$  that is hidden from the user. The second one being the assumption that the state of the hidden process satisfies the Markov property that is the current state depends solely on the previous state and none of the states before that.

## 2 Related Work

We have many deep learning models detecting the text based on available corpus by providing input text and grammar corrections. However, these networks have fallen out of favour for modelling sequential text data, as they require context lengths, more computation, more data and previous hidden state summary of different time stamps. In the neural networks [4] approach which chooses different entities to predict next words, neural networks has to be trained and data is generated based on previous entity provided which results in additional memory being consumed. This method performs something similar to teacher forcing models. These models are trained by feeding the ground truth words to the network for generating the different parts of the sentence.

To avoid this we have popular variations of Recurrent Neural network models such as long short-term memory (LSTM) and Gated recurrent unit (GRU) where text generation happens based on usage of words and its probable occurrence from generated sentence. In such models, all possible words are predicted and appended for forming sentences, reassessment will be performed to results later. Though possible sentence generation is high, evaluating all the possible sentences takes more computational time and memory.

In models where Generative Adversarial Networks (GANs) are used, the generator is trained

to produce high quality samples but accuracy obtained is more for images than for text sequences.

Some of other deep learning models where RNN's are designed to process sequential information with help of previous state i.e. memory. At every processing step, input sequences, accumulating information from past are presented to RNN, which modify network state. LSTM [2] sequence-to-sequence models are a special class of RNN's known for their ability to effectively learn long-term dependencies in sequences. These models maintain a forget gate, which determines how much of previous cell state should be passed on current time stamp. Sequence to sequence models are applied in video captioning, speech recognition etc. These models use encoder-utilizing words of source sentence in forming context vector, which summarizes semantics of a sentence and decoder for operating on this semantics vector to generate required translation words. These models give less execution efficiency and are compute intense.

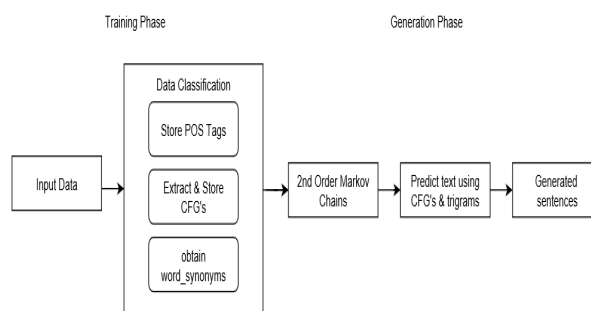
As deep neural network models are very dense in computing [1], we have popular methods of analysis. Hidden Markov Model (HMM) [3] which estimates the probability of text occurrence in given position, based on sequence of preceding values. Number of occurrences of words length  $(k+1)$  in learning vector is calculated. Transition matrix approach is used for getting probable occurrences of data under different conditions. For smaller values of  $k$  this approach is efficient but, as  $k$  grows transition matrix also grows and this will cause the problem of insufficient memory to store the vectors.

Context Free grammar (CFG) [5] tool kits are also available which generated based on usefulness and reachability of text. Programming languages laboratory at university of Caligari provided an online tool for context Free grammar checker to check basic properties of context free grammars, where in the tool generates not more than 20 sentences, which are the first ones ordered by sentence length. Generated sentences are too simple. Some of the best CFG tools are used in several research projects like SAQ and grammar testing methods where Purdom's algorithm and CDRC-P algorithms are used but

still failed in generating appropriate text sentences. Our approach is quite simple, we combine CFG's and HMM models where context free grammars are generated from the input sentences and the structures are stored, which results in higher accuracy and meaningful sentences for the user.

### 3 Proposed Methodology

We propose the use of CFG's to understand the structure of the sentences from the input data and use HMM to predict the words based on the CFG. We use both of them together in tandem to perform a selective prediction of words. The prediction is a two-step process, we use the CFG's to identify the type of word that will be predicted next (such as ADJ, VERB, PROPEN etc.) and then with the help of a second order markov chain we make a prediction of the next word that is of the type expected in the CFG.



**Fig. 1.** Proposed methodology to generate sentences

For example if we have to generate sentences for the CFG [ADJ, NOUN, PUNCT]. As we are using a second order markov chain we choose the first two words randomly from the part of speech (POS) as mentioned in the CFG. In this case we choose words from the part of speech type of ADJ and NOUN. While selecting the first two words we should make sure that it is a part of a valid second order markov chain (will be explained later

**Algorithm 1: Procedure for sentence generation**


---

```

POS tagging on input data
POS[tag] = word_list
obtain markov_chains from input data
obtain CFG_sentence_structure from input data
for each CFG in CFG_sentence_structure
    final_sentence = ""
    w1 = random_word(POS[CFG[0]])
    w2 = random_word(POS[CFG[1]])
    while (w1,w2) not in keys(markov_chains) :
        w1 = random_word(POS[CFG[0]])
        w2 = random_word(POS[CFG[1]])
    for i from 2 to len(CFG)
        if markov_chains[(w1,w2)] intersection POS[CFG[i]]
                                                    exists
            next_word = random_word(POS[CFG[i]])
            while next_word not in markov_chains[(w1,w2)]
                next_word = random_word(POS[CFG[i]])
            final_sentence += " " + next_word
            w1 = w2
            w2 = next_word
    print final_sentence

```

---

**Fig. 2.** Algorithm that is used to generate sentences

in detail). So in this case, assuming we choose the words 'Good' and 'Morning', as it is part of a valid second order markov chain, we proceed. The next word predicted should be of the type PUNCT and it should be preceded by 'Good Morning'. We randomly choose the next word from the second order markov chains, assuming the selected word is '!'. As we have reached the end of the CFG the sentence generation is over and the final sentence output is 'Good Morning!'.

To provide more variety to the generated sentences, synonyms of the predicted word are placed in the generated string. Once the word is predicted, synonyms of that particular word are obtained and put into the generated sentence. As synonyms have the similar meaning to the original predicted word even if we do substitute the actual word with the synonym word the semantics of the generated message will not vary.

This research work presents two different phases, the **training phase** and the **generation phase**. The training phase involves the preparation of the data and the creation of all the required components for the generation phase. These include the POS tags, the CFG list etc.

**3.1 Training Phase**

In this phase we extract all the necessary details from the input data and then store them for further use in the generation phase. The first step in this phase is the POS tagging of the input data. This is important as it would help in understanding the structures of the sentence in the input data and also would get rid of ambiguity by realising all the parts that each word takes, for example the word 'sun' can either be a verb or a noun. Part of speech also helps us understand what the word means in that particular context.

So the first step is tag the input data to with the POS tags, this is done with the help of the spacy tool. Spacy internally tokenizes the text and gives the POS tags to each token. For the tweets input data, about 13 unique parts of speech have been identified and used for prediction. These POS tags are used for two different tasks. Firstly they are used to store all the words for the 13 unique identified parts of speech and secondly to create the CFG's for all the sentences in the input data.

The first use of the POS tagging is to store all the words and their respective tags. A dictionary is created where the keys are the 13 unique parts of speech and the values are all the words having the respective part of speech. This dictionary will be used in the selective prediction of words, that is the next selected word is not completely random and that it will be guided by the CFG's and this particular dictionary.

The second use of the POS tagging is to obtain all the sentence structures from the input data. As we take the assumption that the input data has the right sentence structure, we will retain the same sentence structure and use it to generate sentences. Each sentence from the input data comprises of one right sentence structure and all of them are stored. Each CFG/sentence structure will be used to generate sentences and they are one of the main requirements as the predictions are based on these structures. Only unique sentence structures are used for the prediction. These two steps consist of the training phase of this work.

### 3.2 Generation Phase

This is the phase where the actual word prediction happens. The input data is used to store the second order markov chains. Second Order Chains are used mainly because they provide the right amount of variations in the generated sentences and also makes sure that the generated sentences are not truly random. Choosing a smaller chain causes the generated sentence to be truly random and choosing a longer chain will reduce the variety by constricting the generation to a specific set of words.

These chains help us in choosing on what the sentence begins with, also as the input data already contains the right lemma of the word, we will reuse this when we predict words. So we will have the right lemma in the right context when we predict words. Having the right lemma of words makes sure that the sentence formation and more importantly the sentence sounds right. At this point we have the list of CFG's, POS dictionaries, and the second order markov chains, that is all of the key items required for prediction.

For each CFG there exists a start state. This start state marks how the sentence begins. In our case the start state is nothing but a part of speech. So we randomly pick a word ( $w_1$ ) from the POS dictionaries for the start state and then build from it. As we are using second order markov chains we repeat the same step that is we randomly obtain another word ( $w_2$ ) as per the second part of speech in the CFG. If the pair [ $w_1$ ,  $w_2$ ] doesn't have any word following it which is the part of speech of the next position in the CFG, then we reselect  $w_1$  and  $w_2$  and move forward. If it does have one or more words following [ $w_1$ ,  $w_2$ ] that have the part of speech as expected in the CFG, we randomly select until we obtain a word ( $w_3$ ) that is of the expected part of speech and then append that word to the generated string.

As we are using a second order markov chain the searching key changes from [ $w_1$ ,  $w_2$ ] to [ $w_2$ ,  $w_3$ ] and then the process continues. In case there are no words following the pair [ $w_2$ ,  $w_3$ ] that is of the expected part of speech then we just ignore that part of speech and move to the next expected part of speech. Once we predict a

word, to provide more variety in the strings that are generated we have synonyms for the words. We once again randomly pick a synonym each time and that particular synonym is appended to the final generated string. As the synonyms have the similar lemma and have the similar meaning, replacing them with the actual predicted word will not cause a huge change in the semantics of the generated sentence or the structure of the sentence.

### 3.3 Dataset

The dataset that was used was the Good morning Tweets Dataset. This dataset consists of all the tweets that contain the phrase 'Good Morning'. The dataset consists of about 3000 tweets. The tweets were formatted by removing the retweets and the duplicates. The data was further formatted by getting rid of the url links. All the hashtags were removed as they are not useful in this particular task. The dataset is available online[6]

## 4 Evaluation Criteria

We use two different evaluation criteria. The first one is a score very similar to that of the Bilingual Evaluation Understudy (BLEU) score. In this scoring method, from the generated text we calculated the number of second order markov chains that are present from the initial input data. All second order markov chains are obtained from the generated dataset and counted if that chain is present in the input data. If it is present then we know that the predicted text has the right lemmas for the word and that the context will mostly be maintained, so we count it as a right chain, else it is identified as a bad chain.

One side effect we noticed from this was that, in the predicted text we had synonyms instead of the actual predicted words. As some of the synonyms were not present in the initial vocabulary of words, it was giving a false score in some of the cases but it's not a wrong thing because that sequence has similar meaning but is being wrongly penalised. So to overcome this effect we replace all the key words with the actual word that had been predicted.

As one word can be part of multiple synonyms, there will be a clash as to through which predicted word we got the particular synonym. To solve this problem we store each synonym with a particular version (such as 1.0 or 1.1) to depict which predicted word the synonym had come. So based on the version of the synonym we replace it with the respective predicted word. And then go forward with this criteria:

$$Precision = \frac{n_v}{n_a}, \quad (1)$$

where,  $n_v$  = number of valid chains in generated sentences,  $n_a$  = number of chains in generated sentences.

As seen from above, the score is the number of matching second order markov chains divided by the total number of markov chains.

The second criteria is a subjective evaluation of the generated sentences. A 1000 sentences were randomly selected from the generated sentences. Each sentence was evaluated on 3 main key points:

- Sentence structure,
- Vocabulary in the sentence,
- Semantic meaning preservation.

Sentence Structure means to check if the general structure of the sentence is maintained or not. We check how the generated sentence is arranged grammatically, that is evaluate if the parts of speech are placed in the right part of the generated sentence.

Vocabulary is the second stage of checking. In this we check the usage of the words. Also the check of the right lemma in the different contexts verifies that the vocabulary in the generated sentence is right.

The third and final check in this evaluation criteria is the revival of the semantic meaning from the input data. This is to check if the semantic meaning of the generated sentence is similar to the one from the input sentence. Here the generated sentence is compared with the ground truth and if they do have similar semantics then it is considered as a good generated sentence.

All the above three criteria are based on comparison with the ground truth. If sentence satisfies all the above criteria then it was counted as a valid sentence.

## 5 Results

The evaluations were done on five different models. The five models are as given below:

- The first model generated sentences using only a Context Free Grammar.
- The second model generated using a Hidden Markov Model.
- The third one was another implementation which uses the above two technologies.
- The fourth one was our implementation which combines both CFG's and HMM's.
- The fifth one being a LSTM Model.

As stated above we use two methods for evaluation. The first one is a score similar to a BLEU score. The second method is a subjective evaluation.

In the first method we create all second order markov chains from the generated sentences and observe how many of these chains have been observed before. With this method we have observed a precision of 83%.

**Table 1.** These results depict the amount of second order markov chains that have been retained from the ground truth. Results are shown for all five models where the fourth model is our implementation

Dataset	Technique	Chains Retained
Good Morning	CFG	25.9 %
Tweets	HMM	<b>86.2 %</b>
	CFG & HMM	30.5 %
	<b>CFG &amp; HMM II</b>	<b>83.7 %</b>
	LSTM	28.8 %

As seen from Table 1 a comparison was made between the number of second order markov

chains that were retained from the original dataset. The Hidden Markov Model performs the best as it works solely on the second order markov chains and then comes our implementation. It performs much better than the alternative implementation that also uses both HMM and CFG to predict the text.

**Table 2.** Subjective evaluation was performed on the five different models. The fourth model is our implementation

Dataset	Technique	Score
Good Morning	CFG	32.3 %
Tweets	HMM	35.5 %
	CFG & HMM	40.7 %
	<b>CFG &amp; HMM II</b>	<b>52.2 %</b>
	LSTM	49.8 %

The second method we used was a subjective evaluation. As seen from the table 2, our implementation performs much better than the other methods. This is a result of combining two methods that perform fairly poorly and to produce a model that performs much better. When we combine the two methods we are utilising the advantages of each model in our own model. The biggest advantage is that this model will work with less input data.

## 6 Conclusion

Neural Network models perform really well, but are compute intensive and require a large amount of data. If there is no large amount of data then they do not perform well. If there is no computation power then these class of algorithms go for a toss. The problem of less data for training can be handled using CFG and HMM to predict text. But individually they do not perform well, but when combined together they perform much better. The next good part is that not a lot of computation power is required in the proposed method. It was observed when combined together, the proposed method was able to retain the semantic meaning, the grammatical structure and sentence structure from the original sentences.

The proposed method has a low memory footprint as not all possible sentences are generated and then evaluated. In this case we just pick one of suitable words based on the context and the structure. In the earlier methods all possible words are appended to the string rather than word, in this way the proposed method has a low footprint. In this case the method does not have to go through the problems of remembering data from the past as it is involved with generating simple sentences not based on any data from the past.

The proposed work can be used in the cases of greeting the user in a unique way each time, or in the case of giving an automated reply, or when the system wants to remind the user to do something. When the variety is given to the user then it makes the user feel good about the system that they have. This work has shown that the proposed method can achieve significant results and can be used in the above mentioned methods.

Going forward, the proposed novel method can be combined with deep learning techniques so that we can attain the accuracy that is achieved (by Neural Networks) with lesser data and as well as a lower memory footprint. We can take the positive things from both the models and work on coupling them together and obtaining a model that can achieve very high accuracies.

## References

1. Fedus, W., Goodfellow, I., & Dai, A. M. (2018). Maskgan: better text generation via filling in the... *arXiv preprint arXiv:1801.07736*.
2. Sen, S. & Raghunathan, A. (2018). Approximate computing for long short term memory (lstm) neural networks. *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, Vol. 37, No. 11, pp. 2266–2276.
3. Szymanski, G. & Ciota, Z. (2002). Hidden markov models suitable for text generation. *WSEAS International Conference on Signal, Speech and Image Processing (WSEAS ICOSIP 2002)*, pp. 3081–3084.
4. Xie, Z. (2017). Neural text generation: A practical guide. *CoRR*, Vol. abs/1711.09534.

5. Xu, Z., Zheng, L., & Chen, H. (2010). A toolkit for generating sentences from context-free grammars. *2010 8th IEEE International Conference on Software Engineering and Formal Methods*, IEEE, pp. 118–122.

*Article received on 18/01/2019; accepted on 04/03/2019.  
Corresponding author is Samarth Navali.*



# Sentence Similarity Techniques for Short vs Variable Length Text using Word Embeddings

D. Shashavali, V. Vishwjeet, Rahul Kumar, Gaurav Mathur,  
Nikhil Nihal, Siddhartha Mukherjee, Suresh Venkanagouda Patil

Samsung R & D Bangalore,  
India

{shasha.d, v.vishwjeet, rahul.k4, gaurav.m4,  
nikhil.nihal, siddhartha.m, suresh.patil}@samsung.com

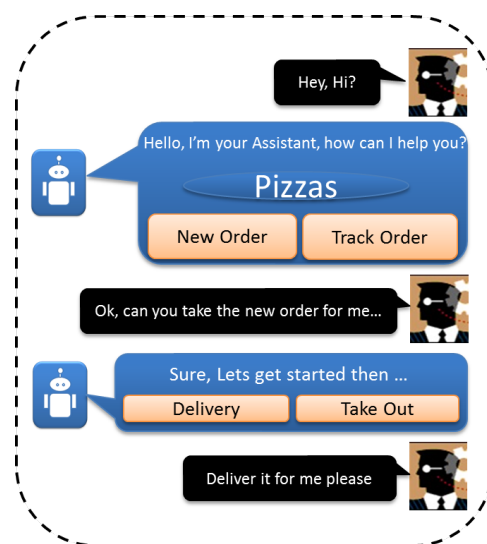
**Abstract.** In goal-oriented conversational agents like Chatbots, finding the similarity between user input and representative text result is a big challenge. Generally, the conversational agent developers tend to provide a minimal number of utterances per intent, which makes the classification task difficult. The problem becomes more complex when the length of the representative text per action is short and the length of the user input is long. We propose a methodology that derives Sentence Similarity score based on N-gram and Sliding Window and uses the FastText Word Embeddings technique which outperforms the current state-of-the-art Sentence Similarity results. We are also publishing a dataset on the shopping domain, to build conversational agents. And the extensive experiments done on the dataset fetched better results in accuracy, precision and recall by 6%, 2% and 80% respectively. It also evinces that our solution generalizes well on the low corpus and requires no training.

**Keywords.** Sentence similarity, word embeddings, natural language processing, sliding window, N-grams, text classification.

## 1 Introduction

The determination of sentence similarity in natural language processing has a wide range of applications. In applications like Chatbots, the uses of sentence similarity include estimating the semantic meaning between the user input and button text. Hence, such applications need to have a robust algorithm to estimate the sentence similarity which can be used across a variety of

domains. Well, the main reason we want to infer meaning from raw text is that NLU aims at building systems that understand user utterance and trigger meaningful results based on the user input. Refer Figure 1 for example.



**Fig. 1.** The primary goal of the dialogue systems is to understand the user's input or goal by using NLU techniques, the bot must manage to achieve a goal by showing the appropriate action

Multitask learning [13] schemes along with supervised and unsupervised approaches have lead to the betterment of NLP task results.

A simple approach [6] using WMD (Word Mover's Distance), which measures the dissim-

ilarity between two sentences, as the minimum distance that the embedded words of a sentence need to travel to reach the embedded words of other. The recent approach [14] to sentence level semantic similarity technique is based on unsupervised learning from conversational data. This approach process the sentences in a high dimensional space and doesn't fetch better results on short sentences, so it's very hard to learn direct Sentence Embeddings. Also, the most recent Sentence Encoder models [4], Transformer encoder and Deep Averaging Network (DAN) have a trade-off of accuracy and computational resource requirement. Moreover, one needs to build the deep neural networks (DNN) or more sophisticated architectures and train the model with the large corpus.

Here, we propose methods which are based on Cosine similarity calculation along with Sliding window and Weighted N-gram. The proposed approach is fairly simple in architecture and outperforms the latest Universal Sentence Encoder technique [4].

## 2 Related Work

Sentence similarity has many interesting applications such conversational agent with script strategies [1] and the Internet. The recent work in the area of natural language processing has contributed valuable solutions to calculate the semantic similarity between words and sentences. However, much research has been done on measuring long text similarity, the computation of sentence similarity is far from perfect [7, 5, 8]. We propose to compute sentence similarity between a very short (1-3 words) and lengthy sentences. Bag of word cosine similarity does not take care of word order in a sentence. For example, "Do I not look good?" and "I do not look good." will have a 100% cosine similarity score. For document similarity, weighted N-Gram over cosine similarity is being suggested in 3.2.2. We took N-Gram weighting formula from the paper [3].

The use of unsupervised word embedding representation of words as vectors, is to preserve semantic information [10]. The Wordwise sum of vectors or average of the vectors also produces a

vector with the potential to encode meaning. The mean was used as baseline in [11]. The sum of word embeddings first considered in [10] for short phrases, was found to be an effective model for summarization in [9].

The cosine distance, as is commonly used when comparing distances between embeddings, is invariant between sum and mean of word embeddings. Both sum and mean of word embeddings are computationally inexpensive, given the fact that pre-trained word embeddings are available. Deep learning solutions [12] handle sentence similarity with variable-length but, requires a huge chunk of data to train and is resource heavy to train and maintain.

## 3 Model Architecture

The proposed methodologies use Word Embeddings and Cosine similarity techniques for word representation and calculating similarity score.

### 3.1 Word Embedding and Cosine Stacks

**Word Embeddings.** Word embeddings computed using diverse methods are basic building blocks for Natural Language Processing (NLP) and Information Retrieval (IR). They capture the similarities between words [2]. And as our approach is naturally dependent on a word embedding, we've chosen FastText [3] over other embeddings. Firstly, subword information is taken into consideration in which each word  $w$  is represented as a bag of character N-gram. This further signifies that, for previously unseen words (e.g. due to typos), the model can make an educated guess towards its meaning, thus allowing to learn reliable representation for rare words. Inherently, this also allows you to capture meaning for suffixes/prefixes. Second, and most importantly, we notice that the proposed approach provides very good word vectors even when using small training datasets.

**Cosine Similarity.** The cosine similarity between two vectors (or two sentences on the Vector Space) is a measure to calculate the cosine of the angle between them. This metric is a measurement of orientation and not magnitude. It

can be seen as a comparison between sentences on a normalized space because, we're not only taking into consideration the magnitude of each word count (tf-idf) of each document, but also the angle between the sentences. To obtain the equation for cosine similarity, we simply rearrange the equation of dot product between two vectors.

$$\begin{aligned}\vec{a} \cdot \vec{b} &= |\vec{a}| |\vec{b}| \cos \Theta, \\ \cos \Theta &= \frac{\vec{a} \cdot \vec{b}}{\|\vec{a}\| \|\vec{b}\|}.\end{aligned}\quad (1)$$

### 3.2 Approaches

We detail two different methods which are as follows.

#### 3.2.1 Sliding Window with Average Weighted Word Vectors

In language, the meaning of the sentence is reflected by the words in it. Older methods used the weighted average of word embedding to represent the sentence and cosine similarity. But, as we are comparing the similarity between short and long sentences, doing the weighted average on a long sentence doesn't help. Moreover, it reduces the weight of the main action verb in the overall representation, which in turn affects the sentence similarity. To overcome this, we use the sliding window approach (Fig. 2) on a long sentence, so that the main action verb weight will be the same in both inputs.

After applying sliding window on  $S_2$ , we get a list of substrings  $S_2'$ . For vector representation of every window, we iterate through the  $S_2'$  and take the weighted average of word embedding, to find the cosine similarity with  $S_1$ . The final similarity score for  $S_1$  and  $S_2$  is taken as the maximum score, obtained from the window comparisons. In Chatbot application, False Positive must be very less for better user experience. We tried the weighted N-gram approach to further reduce false positives.

#### 3.2.2 Weighted N-gram Vectors

N-grams are consecutive strings of N words, for example, trigrams are all possible three word long substrings of a given sentence. To compare two sentences, the sentences are tokenized into unigram, bigram and trigram.

For every unigram of sentence  $S_1$ , find similarity with every unigram of sentence  $S_2$  and select the maximum score as match score for that unigram. All the selected unigram scores are averaged over to get a final unigram score:

$$\begin{aligned}score_1 &= \frac{1}{N_1} \sum_{n=1, n'=1}^{N_1 N_2} \max_n (similarity(S_1 U_n, S_2 U_{n'})), \\ N_1 &= \text{number of unigrams in } S_1, \\ N_2 &= \text{number of unigrams in } S_2.\end{aligned}$$

Likewise, for every bigram of  $S_1$ , find similarity with every bigram of  $S_2$  and select maximum score as match for that bigram. All the selected bigram scores are averaged over to get a final bigram score:

$$\begin{aligned}score_2 &= \frac{1}{N_1} \sum_{n=1, n'=1}^{N_1 N_2} \max_n (similarity(S_1 B_n, S_2 B_{n'})), \\ N_1 &= \text{number of bigrams in } S_1, \\ N_2 &= \text{number of bigrams in } S_2.\end{aligned}$$

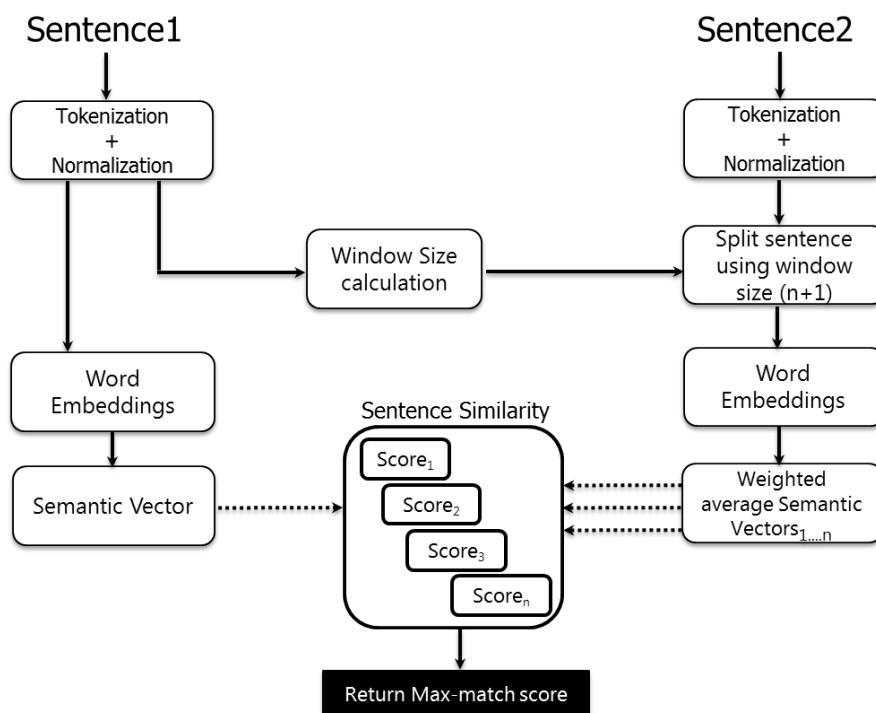
The final similarity score of the sentences is taken as the weighted sum of the final similarity scores of unigrams, bigrams and trigrams:

$$sentence\ similarity = \sum_{g=1}^G w_g * score_g,$$

where

$$w_g = \frac{g}{\sum_{g=1}^G g}.$$

As discussed in section 3.1, we used cosine similarity on averaged word embedding to calculate similarity between N-grams.



❖ Sentence1 is of short-length and Sentence2 is of varied-length

Fig. 2. Sliding window approach

## 4 Results

Here, we describe the data set, which is a conversational data found in Chabot builder based NLP engine environment. We then compare the 3.1 and 3.2 sections with latest Google's Universal Sentence encoder based sentence similarity approach.

### 4.1 Dataset

Although, many datasets are accessible, there are currently no suitable benchmarks (or even standard text sets) for the evaluation of similarity between long and short sentences. We release a dataset<sup>1</sup> which is very specific to conversational agents problem statement. Here, the dataset has been structured into two columns, first, the long sentence which imitates user input and second,

<sup>1</sup><https://github.com/shashavali-d/SentenceSimilarity>

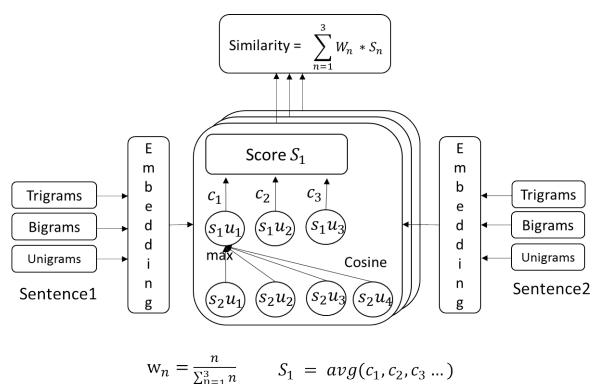
the short sentence which typically resembles the button text in the chat conversation.

Table 1. Sample test dataset

User Input	Button Text
Scrap my order	
Junk my order	Cancel Order
Drop my order	
Display recently viewed items	
Open items I just viewed	Show recent items
Show my last seem items	

### 4.2 Sentence Similarity

A testing instance is a pair of button text and user input. The similarity score between each user input and button text is calculated. Based on similarity score, comparison is categorized as positive or negative. Comparison between button



**Fig. 3.** Weighted N-gram approach

**Table 2.** Results with our approaches vs Universal Sentence Encoder

Approaches/ Metrics	Google Universal Sentence Encoder	Sliding Window with avg. Weighted Vectors	Weighted N-gram Vectors
Recall	0.0789	0.2593	0.9408
Precision	0.9022	0.6507	0.9226
F1 Score	0.1451	0.3708	0.9316
Accuracy	0.9256	0.9298	0.9880

text and user input is deemed positive, if the similarity score is above threshold (0.9). Similarly, the comparison between button text and user input is deemed negative, if the similarity score is below the threshold (0.9). We used the performance metric precision, F1 Score and recall for evaluating our solution.

Our model outperformed Google's sentence similarity in F1 and Recall, see Table 2.

## 5 Conclusion

In the development stages of Chatbots, the current bot platforms provided ML solutions and required large training data from developers. And, the platform had to manage multiple data perpetually and the process became complex and expensive to train the model every time.

In this paper, we propose the sliding window with average weighted word vectors and Weighted N-gram vectors for developing the input semantics vector. The proposed method replaces the sentence embedding approach with simple word embedding based sentence representation and also it doesn't need large dataset for training.

We are excited about the execution of our approaches and will apply the same to other text classification tasks in the near future. We plan to improve the word representation using dependency and constituency parsing information and also, to apply other vector Similarity method than cosine, for the betterment of results.

## References

1. Allen, J. (1988). *Natural Language Understanding*. Benjamin-Cummings Publishing Co., Inc., Redwood City, CA, USA.
2. Bengio, Y., Ducharme, R., Vincent, P., & Janvin, C. (2003). A neural probabilistic language model. *J. Mach. Learn. Res.*, Vol. 3, pp. 1137–1155.
3. Bojanowski, P., Grave, E., Joulin, A., & Mikolov, T. (2016). Enriching word vectors with subword information. *CoRR*, Vol. abs/1607.04606.
4. Cer, D., Yang, Y., Kong, S., Hua, N., Limtiaco, N., John, R. S., Constant, N., Guajardo-Cespedes, M., Yuan, S., Tar, C., Sung, Y., Strope, B., & Kurzweil, R. (2018). Universal sentence encoder. *CoRR*, Vol. abs/1803.11175.
5. Hatzivassiloglou, V. & Wiebe, J. M. (2000). Effects of adjective orientation and gradability on sentence subjectivity. *Proceedings of the 18th Conference on Computational Linguistics - Volume 1, COLING '00*, Association for Computational Linguistics, Stroudsburg, PA, USA, pp. 299–305.
6. Kusner, M. J., Sun, Y., Kolkin, N. I., & Weinberger, K. Q. (2015). From word embeddings to document distances. *ICML*.
7. Landauer, T. K., Foltz, P. W., & Laham, D. (1998). An introduction to latent semantic analysis. *Discourse Processes*, Vol. 25, No. 2-3, pp. 259–284.
8. Landauer, T. K., Laham, D., Rehder, B., & Schreiner, M. E. (1991). How well can passage meaning be derived without using word order: A comparison of latent semantic analysis and humans. *Proc. of the 19th annual meeting of the*

- Cognitive Science Society*, Erlbaum, Mahwah, NJ, pp. 412–417.
9. **Le, Q. V. & Mikolov, T. (2014).** Distributed representations of sentences and documents. *CoRR*, Vol. abs/1405.4053.
  10. **Mikolov, T., Sutskever, I., Chen, K., Corrado, G., & Dean, J. (2013).** Distributed representations of words and phrases and their compositionality. *CoRR*, Vol. abs/1310.4546.
  11. **Pennington, J., Socher, R., & Manning, C. D. (2014).** Glove: Global vectors for word representation. *Empirical Methods in Natural Language Processing (EMNLP)*, pp. 1532–1543.
  12. **Ramaprabha, J., Das, S., & Mukerjee, P. (2018).** Survey on sentence similarity evaluation using deep learning. *Journal of Physics: Conference Series*, Vol. 1000, pp. 012070.
  13. **Subramanian, S., Trischler, A., Bengio, Y., & Pal, C. J. (2018).** Learning general purpose distributed sentence representations via large scale multi-task learning. *International Conference on Learning Representations*.
  14. **Yang, Y., Yuan, S., Cer, D., Kong, S., Constant, N., Pilar, P., Ge, H., Sung, Y., Strope, B., & Kurzweil, R. (2018).** Learning semantic textual similarity from conversations. *CoRR*, Vol. abs/1804.07754.

Article received on 26/02/2019; accepted on 04/03/2019.  
Corresponding author is D. Shashavali.

# Similarity Driven Unsupervised Learning for Materials Science Terminology Extraction

Sapan Shah, Sarath S, Sreedhar Reddy

TRDDC, TCS Innovation Labs, Pune,  
India

{sapan.hs, sarath.s8@, sreedhar.reddy}@tcs.com

**Abstract.** Knowledge of material properties, microstructure, underlying material composition and manufacturing process parameters that the material has undergone is of significant interest to materials scientists and engineers. A large amount of information of this nature is present in the form of unstructured sources. To access the right information for a given problem at hand, various domain specific search systems have been developed. Domain terminologies, when available, can significantly improve the quality of such systems. In this paper, we propose a novel similarity driven learning approach for automatic terminology extraction for materials science domain. It first uses various intra-domain and inter-domain unsupervised corpus level features to score and rank candidate terminologies. For inter-domain features, we use British National Corpus (BNC) as the general purpose corpus. The ranked candidate terms are then used to generate training data for learning a similarity based scoring function. The parameters of this scoring function are learnt using a Siamese neural network which uses word embeddings learnt from both the domain as well as the general purpose corpora to leverage contrasting term features. The proposed similarity based learning approach consistently outperforms other reported classification approaches on the materials dataset.

**Keywords.** Terminology extraction, computational terminology, domain specific search, natural language processing.

## 1 Introduction

A material's properties depend not only on the chemical composition of the material, but also on its internal structure. The structure in turn depends on the processes performed on the

material. Knowledge of composition-process-structure-property relationships is therefore central to the success of materials engineering. A large body of knowledge of this kind is available in the form of publications, company reports, and so on, that capture results from experiments and simulations.

However, finding the right information from this large body that is relevant for a given problem is not an easy task. First, one has to sift through and select right set of documents. Then one has to scan through these documents to extract pieces of information that are relevant to the problem. Traditional search engines are not very helpful here as they are keyword centric and weak on relation processing [15, 16].

Suppose an engineer wants to know what composition of steel gives him a minimum hardness of 40RC when the annealing temperature is in the range of 500-600°C. A simple keyword based search for "steel and composition and hardness 40RC and annealing temperature 500-600°C" will not be very helpful. It will simply retrieve all the documents where the terms steel, composition, hardness, annealing, temperature, 40, 500, 600 appear somewhere in the document without necessarily being related. For instance, 50 need not be related to hardness and 500 need not be related to annealing temperature, resulting in lot of noise. What we need is an intelligent search engine that understands value relations.

To address this need, various domain specific search systems have been proposed in the literature [15, 20]. These systems are not

just keyword centric but also understand domain entities and relations to improve search accuracy. In materials science domain, [20] have developed a system that supports value constraint queries on materials entities. For example, one can use the query “steel & composition & annealing temperature:[500,600]°C & hardness  $\geq$  40RC” for the case discussed above. The search engine looks for mentions of domain concepts in the text and extracts and indexes these mentions and relations between them. It also extracts values associated with the mentions and indexes them.

The generated index is then used for processing user queries. The search engine uses domain dictionaries to identify domain concepts of interest. These dictionaries are usually supplied by domain experts. However, a domain such as materials science is large and continuously growing, so expecting users to supply complete and up to date dictionaries is impractical. Tools that can automatically mine and extract domain terminologies are of great help in this context as they can serve as building blocks for constructing domain dictionaries [13].

### 1.1 Automatic Domain Terminology Extraction

Various approaches to terminology extraction can broadly be classified into supervised, weakly supervised and unsupervised methods. Supervised approaches cast this as a binary classification problem [8, 5, 24]. However, they need a large amount of labelled data for learning. This is hard to come by for an application domain such as materials science. Weakly supervised approaches on the other hand rely on small labelled data and a large pool of unlabelled data to learn classification models in an iterative manner. For instance, co-training based approaches [4]. However, these approaches suffer from the problem of *semantic drift* [19] wherein if non-domain terms are incorrectly added to the labelled data during earlier iterations, the later iterations are adversely affected and this downgrades the overall quality of the extracted terminologies.

Unsupervised approaches such as [23] primarily depend on various unsupervised corpus level intra-domain and inter-domain base features such

as C-value, TF-IDF, domain relevance, etc. Sophisticated scoring functions are then defined using these base features that try to capture termhood and unithood [11] of various domain terminologies. Co-training based approaches are also proposed in literature where the labelled data is generated from base features in a fully unsupervised manner. This is essentially done by ranking the candidate terms using the scoring function and taking the top  $p$  terms as positive examples and bottom  $p$  as negative examples. The parameter  $p$  is critical to the performance of the classifier. With larger  $p$ , the distinction between positive and negative examples blurs, and with smaller  $p$ , we do not have enough training data.

In this paper, we propose a novel similarity driven learning approach as opposed to standard classification based approaches for unsupervised terminology extraction. In this approach, as against taking top  $p$  terms as positive examples and bottom  $p$  as negative, we take pairs of terms: we pair top  $p$  terms with each other to generate *similar* data set, and we pair top  $p$  terms with bottom  $p$  terms to generate *dissimilar* data set. Thus we have  $p^2$  positive examples and  $p^2$  negative examples, significantly increasing the training data size. This allows us to choose a small enough  $p$  that sharply delineates positive terms from negative terms.

We first use various corpus level statistical features to score and rank candidate terms. We use both intra-domain and inter-domain features for this purpose. For inter-domain features, we use British National Corpus (BNC)<sup>1</sup> as the general purpose corpus. Inter-domain features essentially measure the contrastive nature of domain specific terms. The ranked candidate terms are then used to generate training data for learning a similarity based scoring function. The parameters of the scoring function are learnt using a Siamese neural network [10] that uses word embedding representations of the candidate terms. We use two embeddings to represent a term - one learnt from the domain corpus and the other from the general corpus to leverage contrasting features present in the two corpora.

The proposed Siamese network based method has been compared with standard baselines

<sup>1</sup>available at <http://www.natcorp.ox.ac.uk/>



such as C-value, domain relevance, etc., used in terminology extraction literature. It is also compared with a co-training based unsupervised fault tolerant learning approach proposed by [23]. Our method outperforms both the baselines as well as the co-training approach. To evaluate the effectiveness of similarity driven learning, we also compare our model with a standard feed forward network having similar complexity.

The rest of the paper is organized as follows. Section 2 discusses the relevant related work. Section 3 explains pre-processing steps to identify candidate terminologies as well as the base unsupervised features used by our model. Section 4 describes the similarity driven learning approach and details the learning task for Siamese network based scoring function. Section 5 describes the evaluation dataset and discusses the experimental results. Section 6 summarizes this work and indicates future work directions..

## 2 Related Work

Fault Tolerant Learning (FTL) [23] is a completely unsupervised iterative learning technique for term extraction leveraging ideas from co-training [4] and transfer learning [3]. FTL trains two support vector machine classifiers separately, where predictions from one classifier are verified by the other to improve term extraction performance. Input data for one classifier is generated using the TF-IDF measure whereas the other classifier uses delimiter candidate term extraction. [22] have proposed a weakly supervised co-training based approach where they focus on learning multiple representations for terms by composing constituent words using convolutional neural network and recurrent neural network based classifiers. However, these iterative co-training approaches primarily treat term extraction as a binary classification problem. The method proposed in this paper instead learns a similarity based scoring function that captures feature similarities of domain terms as opposed to discriminating domain terms from non-terms.

A closely related work that combines word embeddings from domain specific as well as general purpose corpora is by [2]. However, the

local-global vector based approach suggested by the authors only considers unigram terminologies and build a pure classification model as opposed to a similarity driven model such as the one proposed in this paper.

## 3 Pre-Processing

Automatic terminology extraction methods first employ various linguistic and statistical filters to identify candidate terminologies. The standard filters used in the literature include Parts of Speech (PoS) tag filter, stop words filter, frequency filter, and so on. Once the set of candidate terms are identified, scores for various unsupervised corpus level intra-domain and inter-domain features are computed. Following describes various features used by our model.

### 3.1 Intra-domain Features

This category of features are important in bringing out the terms that are most frequent within a particular domain. They are primarily statistical in nature. The following is a summary of the intra-domain features used in our model.

- TF-IDF [18]: It is a product of TF (frequency of term within a document) and IDF (Inverse Document Frequency - the number of documents in which a term occurs). For the purpose of term extraction, we take the average TF-IDF values across all documents in the corpus as TF-IDF feature.
- C-value [9]: This unithood feature scores candidate terms using a combination of the following criteria: assigns higher scores to more frequent terms; penalizes candidate terms if they occur as substrings of larger candidate terms; assigns higher scores to longer candidate terms.
- Term Variance (TV) [6]: It scores a candidate term by measuring its variance across all documents in the corpus. It discriminates between high frequency non-terms appearing in all documents from terms that occur frequently in a small set of documents.

### 3.2 Inter-domain Features

The inter-domain features used by our model include,

- Domain relevance [7]: It compares the frequency of candidate terms in domain corpus and general corpus.
- Relevance [7]: It improves domain relevance by down weighting candidate terms that occur rarely in domain corpus or occur highly frequently in general corpus.
- Weirdness [1]: This measure is similar to domain relevance but takes relative frequencies into account by considering dataset sizes.

The features described above are expected to score candidate terms such that true domain terms are assigned higher scores compared to non-terms. Accordingly, candidate terms are ranked in descending order by the score value. This ranked list is then used to measure *precision@k* which computes the number of correct domain terms identified among the top  $k$  candidate terms in the list.

An issue with all these statistical features is that they are very sensitive to term frequency, so they fail to identify terms that lack statistical significance. Hence they need to be augmented with learning based approaches that learn to identify other aspects of similarity to distinguish domain terms from non terms. The ranked list produced by the base features can serve as the starting point to identify a seed list of positive and negative examples to train a classifier.

### 3.3 Pre-trained Embeddings

For the learning phase, we represent the candidate terms using pre-trained word embeddings. Inspired by the local-global vector based approach proposed in [2], we use pre-trained 100 dimensional GloVe [17] vectors to represent statistical strength of words as they appear in general corpus (referred as *general* vectors). Whereas, word embeddings capturing domain semantic similarity are learnt using domain specific text corpus (referred as *domain* vectors). A unigram term in our

system is represented by concatenating its general and domain vectors. The input representation for multiwords then corresponds to concatenation of constituent unigram terms. We currently consider multi grams with maximum size 3. The rest of the paper refers to this representation as *pre-trained* vector.

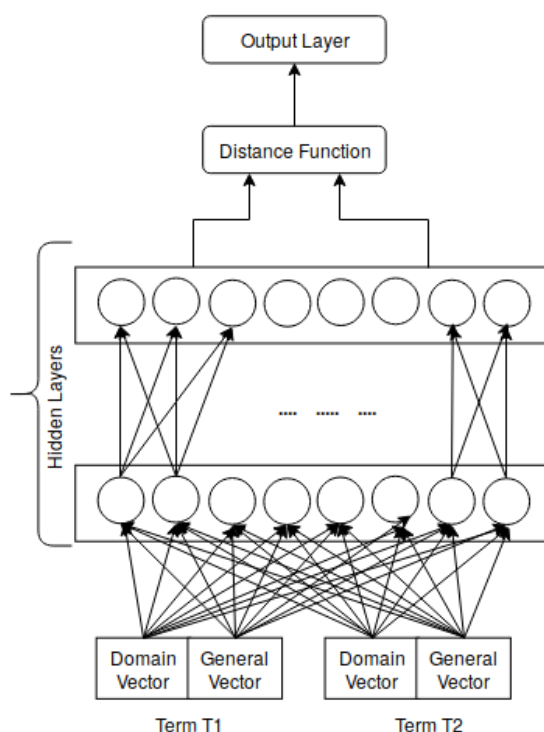
## 4 Proposed Similarity Driven Scoring Function

As mentioned earlier, our approach leverages feature similarity across domain terms to learn a term scoring function. Figure 1 shows the Siamese network architecture (referred as SNet) used to learn this function. The network takes a pair of terms as input and outputs a similarity score between them. Training instances for learning parameters of this network are generated in the following way.

1. The scoring functions explained in section 3 are used to generate a ranked list of candidate terms. Domain terms are expected to be ranked higher in this list.
2. Top  $p$  terms from the ranked list are denoted as *positive terms* whereas bottom  $p$  terms are denoted as *negative terms*.
3. Total  $p \times (p - 1)$  pairs of terms are generated using *positive terms* and assigned similarity score of 1.
4. Total  $p \times p$  pairs of terms are generated by taking the cross product of the terms present in the positive and negative term sets. These pairs are assigned a score of 0.
5. The word pairs generated in step 3 and 4 constitute the training data for learning the parameters of the Siamese network in figure 1.

The data generation framework discussed above assumes that the terms closer to the top of the ranking are likely to be true domain terms and the terms closer to the bottom are likely to be non-domain terms. Hence the selection of the top  $p$  terms as positive terms and the bottom  $p$  as negative terms.

The parameters of the Siamese network are learnt using stochastic gradient descent with various choices for distance functions such as Euclidean distance and Manhattan distance. Once the network parameters are learnt, all remaining candidate terms are scored in the following way: a total of  $p$  term pairs are formed by taking the cross product of the given candidate term with all *positive terms*. These pairs are then passed through the network in figure 1 to compute their similarity scores. The average similarity score across these  $p$  pairs is then used as the score for the candidate term. Once the scores for all candidate terms are computed, they are ranked in descending order to generate a ranked list of domain terms.



**Fig. 1.** Siamese Network architecture for scoring similarity between term pairs

For comparison, we also use a simple feed forward neural network (referred as FFNet) architecture. Top  $p$  terms ranked by the base features are marked as domain terms with output

1 and bottom  $p$  terms are marked as negative terms with output 0. These terms are then used to learn the network parameters. Similar to SNet, this network also takes the concatenated *pre-trained* vector of the candidate terms as input. It then uses binary cross-entropy loss to optimize network parameters. Once the parameters are learnt, all the remaining candidate terms are scored using this network and ranked in descending order to generate a ranked list of domain terms. Note that for the same  $p$  positive and  $p$  negative terms, FFNet has only  $2p$  training examples, whereas SNet has  $2p^2$  examples. Also the SNet approach relies on average similarity with all  $p$  positive terms.

## 5 Experimental Evaluation

Since materials science is the focus of our work, we use a materials science corpus for domain terminology extraction. We use British National Corpus (BNC) as the general purpose corpus.

### 5.1 Dataset

The text corpus used for term extraction consists of 1000 publications downloaded from ISIJ<sup>2</sup> International Journal. This Journal contains publications on fundamental and technological aspects of the properties, structure, characterization, processing, etc. of iron, steel and other related engineering materials. The downloaded publications are in the PDF format. We first convert these PDF files to text using Grobid [14].

Following filters are then applied on the converted text: PoS tag filter:  $((Adj)?(Noun)+)|((Adj|Noun) * (Verb)?);$  stop words filter; frequency filter with minimum term frequency of 10; and shallow stemming that only converts plural forms to singular. This resulted in a total 17000 candidate terms. Few example candidate terms are: *quenching*, *quenching temperature*, *grain size*, *elongation* and *tensile strength*. The filters used for candidate term extraction have been designed by analysing few sample documents in the corpus. We use

<sup>2</sup>The Iron and Steel Institute of Japan - <https://www.jstage.jst.go.jp/browse/isijinternational/-char/en>

the pre-trained domain vectors<sup>3</sup> developed by [12] for materials science domain. Whereas for general vectors, we use pre-trained GloVe [17] embeddings<sup>4</sup>.

## 5.2 Comparison with Previous Work

The unsupervised features described in section 3 serve as the baseline. We then compare the performance of SNet and FFNet with a simple voting algorithm and classification based algorithms such as Fault Tolerant Learning (FTL) [23] and Single Classifier (SC). The voting algorithm [25] simply uses the rankings produced by the base features. The score for a term is computed by summing the inverse of its rank in the participating base features. We use intra-domain and inter-domain features to provide different views of data for FTL approach. It starts with an initial list of  $s$  seed terms to bootstrap the classifiers. It then iteratively adds  $n$  high confidence terms to the seed list until convergence. SC is a non-co-training version of FTL that uses only a single classifier. The results were manually evaluated by three domain experts. We use *Precision@k* as the evaluation metric.

### 5.2.1 Experimental Setting

The SNet architecture for the materials science domain contains a single fully connected hidden layer. A distance function is then applied on the output of the hidden layer followed by a sigmoid activation. Binary cross entropy loss is then minimized using RMSProp stochastic gradient descent (SGD) [21]. The network also applies ReLU activation for the units in the hidden layer along with dropout regularization. Grid search used for hyper parameter tuning consists of: hidden layer units in {6, 8, 10, 12}; distance function in {euclidean distance, manhattan distance}; dataset size parameter  $p$  in {100, 200}; data generation features in {all-features, single best feature from each category namely C-value for intra-domain and

domain relevance for inter-domain}. For data generation, the scores for multiple features are combined by taking the average of their normalized scores. The best features from the two categories are decided by considering their *precision@2000*. The architecture for FFNet also consists of a single hidden layer with similar details except for the number of positive and negative terms. This has been varied among {100, 200, 400}. For SNet, the best hyper parameter combination was found to be 8 hidden units, manhattan distance and all-features for data generation with  $p = 100$ ; for FFNet, it was found to be 8 hidden units and 200 positive and negative terms.

Similarly we have performed grid search for hyper parameters of FTL, SC and voting algorithm. In FTL, the initial seed terms ( $s$ ) are varied among {200, 400, 500, 800, 1000} and the number of terms added in each iteration ( $n$ ) are varied among {20, 50, 80, 100, 150}. Different views for the classifiers are provided by using only the best intra-domain feature for one classifier and the best inter-domain feature for the other. We have also tried using combinations of best features for generating seed term list. SC also uses a similar parameter setting. For voting algorithm, we have tried the following configurations for base features: all features; top 2 features; top 2 intra-domain features; and top-2 inter-domain features.

## 5.3 Results and Discussion

Table 1 shows the results of our experiments. The similarity based model implemented by SNet outperforms voting algorithm, classification models such as FFNet, SC and the co-training approach of FTL. For smaller values of  $k$  such as 200, 500, the base unsupervised features have better accuracy with domain relevance giving the best results. This is to be expected as these are frequency based measures and the terms ranked closer to the top are more likely to be domain terms. However, the terms appearing later in the ranked lists for these features are not reliable. The classification and similarity based models give superior results in this case. Again this is on expected lines as these approaches learn to discern other aspects of similarity among positive and negative terms.

<sup>3</sup>downloaded from <https://github.com/olivettigroup/materials-word-embeddings>

<sup>4</sup>downloaded from <https://nlp.stanford.edu/projects/glove/>

**Table 1.** Evaluation of term extraction approaches using *precision@k*

Method	k=200	k=500	k=1000	k=1500	k=2000	k=3000
<b>Unsupervised Features</b>						
C Value	0.64	0.648	0.639	0.636	0.634	0.613
TFIDF	0.825	0.714	0.64	0.594	0.57	0.549
Term Variance Quality	0.77	0.71	0.654	0.629	0.593	0.577
Domain Relevance	<b>0.89</b>	<b>0.868</b>	0.792	0.749	0.720	0.695
Relevance	0.52	0.55	0.554	0.556	0.551	0.525
Weirdness	0.625	0.614	0.599	0.590	0.566	0.541
<b>Proposed Methods and Previous Works</b>						
Fault Tolerant Learning	0.820	0.830	0.804	0.769	0.725	0.725
Single Classifier	0.840	0.868	0.791	0.772	0.703	0.705
Voting Algorithm	0.885	0.866	0.791	0.748	0.720	0.694
FFNet	0.776	0.818	0.785	0.771	0.767	0.759
SNet	0.761	0.822	<b>0.821</b>	<b>0.815</b>	<b>0.806</b>	<b>0.764</b>
FFNet_dict	0.831	0.832	0.802	0.817	0.781	0.755
SNet_dict	0.856	0.858	0.825	0.815	0.797	0.765

This is evident from the table for values of  $k > 500$ . It should also be noted that SNet consistently beats FFNet with about 3% accuracy improvement even though both models have similar network complexity (in terms of number of parameters). This can be attributed to three reasons:

- SNet's architecture is designed to explicitly learn similarity.
- Instead of relying on a single classification decision, SNet brings in ensemble effect by averaging over the similarity scores computed from the top  $p$  high confidence terms.
- For similar network complexity, due to pairing SNet has a much larger dataset available for learning parameters.

In many practical scenarios, small amount of domain terminologies are often available. For instance, in the form of domain dictionaries or lexicons. These existing domain terminologies can be exploited to improve the terminology extraction algorithms. To study the effect of such lexicon, we created a small dictionary of material properties and manufacturing processes. The terms present in this dictionary are added to the list of *positive*

*terms* as part of dataset creation. Table 1 shows results for the classification (FFNet\_dict) and similarity (SNet\_dict) based models.

For smaller values of  $k$  such as 200, 500, these models perform better than FFNet and SNet. This is due to the fact that the dictionary aided models use true domain terms in addition to the terms suggested by unsupervised features. Due to this, the terms which are similar to lexicon (i.e. material properties and processes) are ranked higher. However, the number of terms representing material properties and processes is finite and not very large. Due to this, terms of various other categories (for instance, microstructural features) appear in the ranked list for higher values of  $k$  making domain lexicon less effective. This is observed in the table for values of  $k > 500$ , where the accuracy of both SNet and FFNet approach SNet\_dict and FFNet\_dict respectively.

## 6 Conclusion and Future Work

This paper proposes a novel similarity driven learning approach for materials science terminology extraction. It uses various unsupervised features to generate training data. A similarity based scoring function is then learnt using Siamese network

architecture. The proposed approach outperforms standard classification as well as co-training approaches on materials dataset. Our future work consists of generating typed dictionaries from these terminologies. We are also planning to improve term extraction further by exploiting compositional nature of multiword terms.

## References

1. Ahmad, K., Gillam, L., & Tostevin, L. (1999). University of surrey participation in trec8: Weirdness indexing for logical document extrapolation and retrieval (wilder). *The Eighth Text REtrieval Conference (TREC-8)*, Gaithersburg, Maryland.
2. Amjadian, E., Inkpen, D., Paribakht, T., & Faez, F. (2016). Local-global vectors to improve unigram terminology extraction. *Proceedings of the 5th International Workshop on Computational Terminology (Computerm2016)*, The COLING 2016 Organizing Committee, Osaka, Japan, pp. 2–11.
3. Ando, R. K. & Zhang, T. (2005). A framework for learning predictive structures from multiple tasks and unlabeled data. *J. Mach. Learn. Res.*, Vol. 6, pp. 1817–1853.
4. Blum, A. & Mitchell, T. (1998). Combining labeled and unlabeled data with co-training. *Proceedings of the Eleventh Annual Conference on Computational Learning Theory, COLT' 98*, ACM, New York, NY, USA, pp. 92–100.
5. Conrado, M., Pardo, T., & Rezende, S. (2013). A machine learning approach to automatic term extraction using a rich feature set. *Proceedings of the 2013 NAACL HLT Student Research Workshop*, Association for Computational Linguistics, Atlanta, Georgia, pp. 16–23.
6. Dhillon, I., Kogan, J., & Nicholas, C. (2004). Chapter 4: Feature selection and document clustering.
7. Fedorenko, D. G., Astrakhantsev, N., & Turdakov, D. (2013). Automatic recognition of domain-specific terms: an experimental evaluation. Vassilieva, N., Turdakov, D., & Ivanov, V., editors, *Proceedings of the Ninth Spring Researchers Colloquium on Databases and Information Systems, Kazan, Russia, May 31, 2013*, volume 1031 of *CEUR Workshop Proceedings*, CEUR-WS.org, pp. 15–23.
8. Foo, J. & Merkel, M. (2010). Using machine learning to perform automatic term recognition. *LREC 2010 Workshop on Methods for automatic acquisition of Language Resources and their evaluation methods*, Valletta, Malta, pp. 49–54.
9. Frantzi, K., Ananiadou, S., & Mima, H. (2000). Automatic recognition of multi-word terms: the c-value/nc-value method. *International Journal on Digital Libraries*, Vol. 3, pp. 115–130.
10. Hadsell, R., Chopra, S., & LeCun, Y. (2006). Dimensionality reduction by learning an invariant mapping. *Proceedings of the 2006 IEEE Computer Society Conference on Computer Vision and Pattern Recognition - Volume 2, CVPR '06*, IEEE Computer Society, Washington, DC, USA, pp. 1735–1742.
11. Kageura, K. & Umino, B. (2001). Methods of automatic term recognition - a review. *Terminology*, Vol. 3.
12. Kim, E. Y., Huang, K., Tomala, A., Matthews, S., Strubell, E., Saunders, A., McCallum, A. L., & Olivetti, E. (2017). Machine-learned and codified synthesis parameters of oxide materials. *Scientific data*.
13. Krauthammer, M. & Nenadic, G. (2004). Term identification in the biomedical literature. *Journal of Biomedical Informatics*, Vol. 37, No. 6, pp. 512–526.
14. Lopez, P. (2009). Grobid: Combining automatic bibliographic data recognition and term extraction for scholarship publications. *Proceedings of the 13th European Conference on Research and Advanced Technology for Digital Libraries, ECDL'09*, Springer-Verlag, Berlin, Heidelberg, pp. 473–474.
15. Magle, T. C. (2015). A review of quetzal: A linguistic search engine for biomedical literature. *American Laboratory*. Accessed: 2018-12-01.
16. McCallum, A., Nigam, K., Rennie, J., & Seymore, K. (1999). A machine learning approach to building domain-specific search engines. *Proceedings of the 16th International Joint Conference on Artificial Intelligence - Volume 2, IJCAI'99*, Morgan Kaufmann Publishers Inc., San Francisco, CA, USA, pp. 662–667.
17. Pennington, J., Socher, R., & Manning, C. D. (2014). Glove: Global vectors for word representation. *EMNLP*.
18. Rajaraman, A. & Ullman, J. D. (2011). *Mining of Massive Datasets*. Cambridge University Press, New York, NY, USA.

19. **Riloff, E. & Jones, R. (1999).** Learning dictionaries for information extraction by multi-level bootstrapping. *Proceedings of the Sixteenth National Conference on Artificial Intelligence and the Eleventh Innovative Applications of Artificial Intelligence Conference Innovative Applications of Artificial Intelligence, AAAI '99/IAAI '99*, American Association for Artificial Intelligence, Menlo Park, CA, USA, pp. 474–479.
20. **Shah, S., Vora, D., Gautham, B. P., & Reddy, S. (2018).** A relation aware search engine for materials science. *Integrating Materials and Manufacturing Innovation*, Vol. 7, No. 1, pp. 1–11.
21. **Tieleman, T. & Hinton, G. (2014).** *RMSprop Gradient Optimization (course slides)*. University of Toronto.
22. **Wang, R., Liu, W., & McDonald, C. (2016).** Featureless domain-specific term extraction with minimal labelled data. *Proceedings of the Australasian Language Technology Association Workshop 2016*, Melbourne, Australia, pp. 103–112.
23. **Yang, Y., Yu, H., Meng, Y., Lu, Y., & Xia, Y. (2010).** Fault-tolerant learning for term extraction. *Proceedings of the 24th Pacific Asia Conference on Language, Information and Computation*, Institute of Digital Enhancement of Cognitive Processing, Waseda University, Tohoku University, Sendai, Japan, pp. 321–330.
24. **Zhang, X., Song, Y., & Fang, A. C. (2010).** Term recognition using conditional random fields. *Proceedings of the 6th International Conference on Natural Language Processing and Knowledge Engineering (NLPKE-2010)*, pp. 1–6.
25. **Zhang, Z., Brewster, C., & Ciravegna, F. (2008).** A comparative evaluation of term recognition algorithms. *Proceedings of The sixth international conference on Language Resources and Evaluation*, pp. 28–31.

Article received on 23/02/2019; accepted on 04/03/2019.  
Corresponding author is Sapan Shah.





# Text Classification using Gated Fusion of n-gram Features and Semantic Features

Ajay Nagar\*, Anmol Bhasin\*, Gaurav Mathur

Samsung R&D Institute India, Bangalore,  
India

{ajay.nagar, anmol.bhasin, gaurav.m4}@samsung.com

**Abstract.** We introduce a novel method for text classification based on gated fusion of n-gram features and semantic features of the text. The parallel CNN network captures the n-gram relation between the words based on the filter size, primarily short distance multi-word relations. Whereas for semantic relationship, universal sentence encoder or BiLSTM is used. Gated fusion is used to combine n-gram and semantic features. The model is evaluated on 4 commonly used benchmark datasets (MR, TREC, AG-News and SUBJ), which includes sentiment analysis and question classification. The proposed method is able to surpass the existing state-of-the-art DNN architectures for text classification on these datasets.

**Keywords.** Text classification, convolutional neural network, universal sentence encoder, BiLSTM.

## 1 Introduction

Deep learning models have revealed amazing results in numerous Natural Language Processing(NLP) tasks such as Neural Machine Translation (NMT) [1], Named Entity Recognition (NER) [2], Text Summarization [3], Text Classification [4] etc. Among these, text classification is one of the important and challenging task in NLP, which aims to assign predefined relevant categories to natural language texts. It is useful in many applications like social media text analysis, sentiment analysis applications, business analysis applications, feedback analysis applications etc. Since, there is no complete set of predefined rules for natural languages, classification algorithms are unable to capture complex semantics of the text.

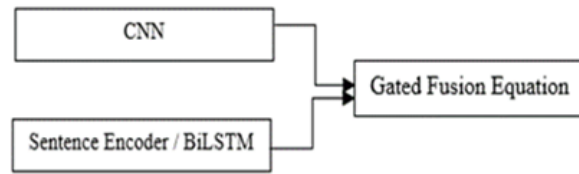
---

\*Equal contribution of authors.

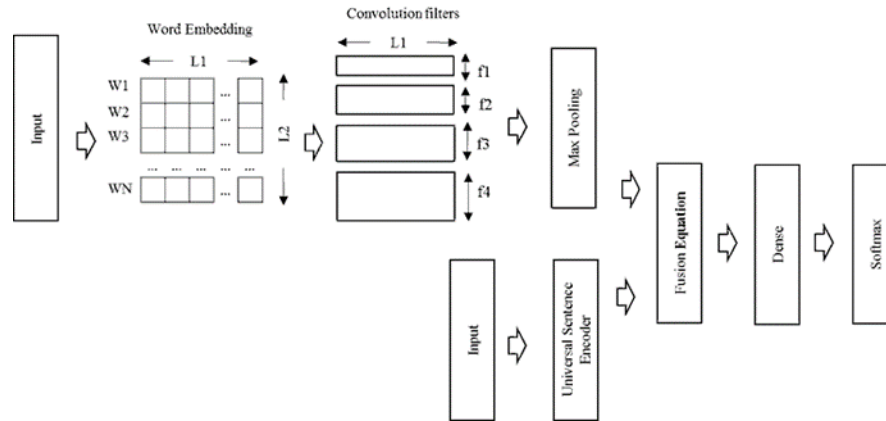
## 1.1 Prior Work

Feature representation for text classification is a crucial problem. Initially, bag-of-words model, which uses unigrams, bigrams, n-grams, were used for feature representation. Later, Mikolov et al. [5] proposed distributed representation of words to solve the data sparsity problem and loss of semantic information of words. Character embedding and sentence embedding are the other types of embedding used for text classification. Word2vec [6] and GloVe [7] are two pre-trained word embedding commonly used for text classification.

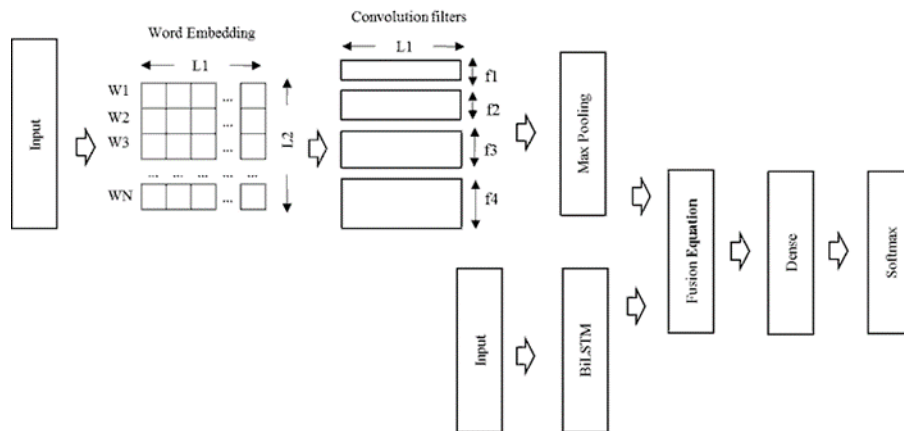
Different deep learning architectures has been applied for text classification to learn different features. Socher et al. [8] proposed the Recursive Neural Network for text classification by modelling sentence representation. Since, text classification problem has sequential nature, Recurrent Neural Networks (RNN) and its variants Gated Recurrent Unit (GRU), Long Short Term Memory (LSTM) were used to learn long distance dependencies or semantics of text. Yoon Kim [4] proposed Convolution Neural Network (CNN) for text classification using pre-trained word embedding as input to learn n-gram features. CNN captures local correlations of spatial or temporal structures but loses the context of text. To take the advantage of both models, Siwei et al., [9] proposed recurrent convolution neural network for text classification to first provide context to each word using RNN and then use CNN to find n-gram features. Zhou et al., [10] also proposed C-LSTM neural network for text classification by applying CNN first and then LSTM.



**Fig.1.** Higher Level Block Diagram



**Fig. 2.** Architecture of using Gated Fusion Equation on CNN and Universal Sentence Encoder



**Fig. 3.** Architecture of using Gated Fusion Equation on CNN and Bidirectional LSTM

Since both architectures used CNN and RNN sequentially, error in CNN network is also propagated to RNN and vice versa. Sequential nature of these architectures may lead to erroneous n-gram features, semantic features or long distance dependencies.

### 1.1 About Our Work

To address the above problem, we propose a novel architecture for text classification using the gated fusion of n-gram features and semantic features.

The intention of this work is robust text classification by modelling n-gram features and long distance dependencies or semantics of text (representation of sentence as a embedding) more concretely. Figure 1 represents higher-level block diagram of our proposed network.

## 2 Proposed Models

As discussed in previous section, both n-gram features and semantic features were considered for classification by taking the advantage of CNN, LSTM and Sentence Encoder. To avoid error propagation from n-gram features extraction to semantic features or vice-a-versa, we trained both networks independently. The architecture loss was minimized based on summation of loss of two models. The CNN was used to capture n-gram features. For long distance dependencies or semantic features, in one model, we used Bidirectional LSTM and in another, we used universal sentence encoder. For fusing both model outputs, we used gated fusion equation. Details of two models are discussed in following section.

### 2.1 Model 1: Gated Fusion on CNN and Bidirectional LSTM

The basic idea is to process the input sentence in two parallel network as shown in the Figure 2. For this, first we tokenize the sentence by splitting at space to get the word sequence. To represent words as distributed dense vectors, we used L1 dimensional GloVe [7] pre-trained word embedding. Unseen words (words not present in GloVe word embedding) were represented as dense vector using uniform random initialization of L1 dimension. For length of input sequence average sentence length L2 was used. In this way word embedding matrix of dimension L2\*L1 for a input sentence was created.

For extracting n-gram features, we used Kim's CNN model [4] as baseline model. Word embedding matrix of a sentence was passed to 4 parallel CNN layers having different filter sizes  $f1*L1$ ,  $f2*L1$ ,  $f3*L1$  and  $f4*L1$ . The filter height was set to  $f1=1$ ,  $f2=2$ ,  $f3=3$  and  $f4=5$ , where height represents number of words to be convolved to capture unigram, bigram and n-grams.

128 filters of each type were taken. After the convolution layer max pooling layer was used to compute most important feature from the output of every convolution. We got 128 features from each convolution layer. These 128 features from every layer were concatenated and a dense vector of size 512 was obtained which constitute n-gram features of a sentence.

Since, LSTMs are able to captures the long distance dependencies in sequential data, we fed word embedding matrix of a sentence to bidirectional LSTM layer to model long distance dependencies or semantic features. We utilized bidirectional LSTM to provide forward and reverse context of the text to the network. We considered 256 hidden units in LSTM. The output of this layer was a dense vector of size 512 (concatenated output of forward LSTM and backward LSTM) which can be interpreted as semantics (long distance dependences) of complete text.

The CNN model output and LSTM model output was given as input to gated fusion equation (Z). The output of gated fusion equation was passed to dropout layer and then to fully connected layer and finally, a softmax layer was used for classification:

$$Z = t \odot g(W_H^1 y^1 + b_H^1) + (1 - t) \odot g(W_H^2 y^2 + b_H^2), \quad (1)$$

$$t = \sigma(W_T y + b_T). \quad (2)$$

Equation (1) represents gated fusion equation, where  $g$  is a nonlinear activation function and for our experiment, we used it as 'relu'. Equation 2, i.e., 't', is called the weightage gate. It represents the weightage given to n-gram features and  $(1-t)$  represents the weightage given to long distance dependencies or semantic features of text. The features generated by the CNN layer and the bidirectional LSTM layer are averaged to generate 'y'. We generated weights by applying sigmoid to 'y' and that are learnable. The intention of using gated fusion was to make model learn to choose itself between features generated by CNN and Bi-LSTM. Since, some texts can be best classified based on short-term dependences and some can be best classified based on long distance dependences, so for model to itself decide the weightage for both these dependences we used gated fusion.

**Table 1.** Benchmark datasets

Dataset	Train Data	Test Data	Classes
MR	10662	CV	2
TREC	5952	500	6
AG News	120000	7600	4
SUBJ	10000	CV	2

**Table 2.** Accuracy on different datasets achieved by our models

Models	Dataset			
	MR	AG News	TREC	SUBJ
Yoon Kim [4]	81.5	86.1	93.6	93.4
CharCNN [14]	77	78.3	76	-
WCCNN [16]	83.8	85.6	91.2	-
KPCN [16]	83.3	88.4	93.5	-
C-LSTM [10]	-	-	94.6	-
BiLSTM-CRF	82.3	-	-	-
F-Dropout [4]	79.1	-	-	93.6
Model 1	79.71 <sup>1</sup>	88.26	93.8	92.611
	80.39 <sup>2</sup>			93.62
Model 2	<b>83.4<sup>1</sup></b>	<b>88.75</b>	95.8	<b>94.951</b>
	84.43 <sup>2</sup>			95.852

For n-gram features, first, we construct the word embedding matrix using input sentence as described for Model 1 and then applied convolution sequence. Kernel sizes, number of kernels, pooling and output dimensions are same as previous model (Model 1).

Similar to previous model, features generated by both the CNN layer and the universal sentence encoder are averaged to obtain  $y$  (Equation 1). The CNN network output and Sentence Encoder output was passed to gated fusion equation (similar to Model 1).

The output of fusion layer was then passed to fully connected layer and finally a softmax layer was used to predict the class. layer, max-pooling layer and flatten layer in Datasets and Experiment Details.

In this section, we first present the benchmark datasets we used for our experiments and then experiment details.

## 2.2 Datasets

We carried out our experiments with 4 benchmark datasets, Table 1 shows the distribution of training and testing data for all four datasets along with number of classes to be predicted.

**MR:** Positive and Negative movie reviews dataset. Aim is to identify a movie review, positive or negative [12]. There is no test dataset defined, so fivefold cross validation (CV) was done.

**TREC:** This dataset contains 6 types of questions. Objective is to identify the class for given question [13].

**AG News:** Topic Classification Dataset. Aim is to classify news into different classes [14].

**SUBJ:** This dataset has sentences and task is to classify a sentence into objective or subjective type [15]. There is no test dataset defined, so for this also fivefold cross validation (CV) was done.

## 2.3 Experiment Details

For all the datasets, training was carried out using mini batch gradient descent with batch size of 64. For binary classification 'binary cross entropy' was used and for other datasets 'categorical cross entropy' loss was being used. We used 'Adam' as an optimizer. No of epoch for each model was taken as 50, though for all the datasets the model converged well below 50 epochs. We used 0.5 dropout rate to reduce overfitting of data.

## 3 Datasets and Experiment Details

In this section, we first present our state-of-the-art results on datasets we used for our experiments and then result discussion.

### 3.1 Experiment Details

We evaluated the proposed methods on different benchmark datasets and compared with state-of-the-art results to show effectiveness of the method. Results of both the models are shown in Table 2. We evaluated the datasets on basis of overall accuracy i.e. correct predictions divided by total number of predictions for test dataset. 5 fold cross-validation is used for cross validation datasets. We observed that using both n-gram features and features generated by universal sentence encoder are able to achieve remarkable results. Using these features, we are able to surpass all the results achieved by multichannel CNN model. Compared to C-LSTM model, we got 1.2% accuracy improvement on TREC dataset. Using n-gram features and features generated by RNN, we are able to achieve approximately same results to other models.

### 3.2 Discussion

It is observed that with the gated fusion of n-gram features by CNN and features generated by RNN, model was able to achieve only the comparable results but with the gated fusion of n-gram features by CNN and semantics by pre-trained universal sentence encoder, model is able to achieve the state-of-the-art results.

Compared with the existing methods, that are using both CNN and RNN for text classification, proposed model performs better. It is noticed that gated fusion can choose between short distance dependency based classification and long distance based dependency classification.

## 4 Conclusion

In this work, we proposed a method for text classification using n-gram features and semantic features captured by convolution neural network (CNN) and universal sentence encoder respectively. Proposed models are able to achieve state-of-the-art results on four common benchmark datasets. The proposed method can be applied to many other natural language processing tasks such as sentence similarity, machine translation (as an encoder) etc.

## References

1. **Bahdanau, D., Cho, K., & Bengio, Y. (2014).** *Neural machine translation by jointly learning to align and translate*. arXiv preprint arXiv:1409.0473, pp. 1–15.
2. **Lee, J.Y. & Dernoncourt, F. (2016).** *Sequential short-text classification with recurrent and convolutional neural networks*. arXiv preprint arXiv:1603.03827, pp. 1–5.
3. **Allahyari, M., Pouriyeh, S., Assefi, M., Safaei, S., Trippe, E.D., Gutierrez, J.B., & Kochut, K. (2017).** *Text summarization techniques: a brief survey*. arXiv preprint arXiv:1707.02268, pp. 1–9.
4. **Kim, Y. (2014).** *Convolutional neural networks for sentence classification*. arXiv preprint arXiv:1408.5882, pp. 1–6.
5. **Mikolov, T., Chen, K., Corrado, G., & Dean, J. (2013).** *Efficient estimation of word representations in vector space*. arXiv preprint arXiv:1301.3781, pp. 1–12.
6. **Mikolov, T., Sutskever, I., Chen, K., Corrado, G. S., & Dean, J. (2013).** *Distributed representations of words and phrases and their compositionality*. *Advances in Neural Information Processing Systems*, pp. 3111–3119.
7. **Pennington, J., Socher, R., & Manning, C. (2014).** *Glove: Global vectors for word representation*. *Proceedings of the 2014 Conference on Empirical*

*Methods in Natural Language Processing (EMNLP)*, pp. 1532–1543.

8. **Socher, R., Perelygin, A., Wu, J., Chuang, J., Manning, C.D., Ng, A., & Potts, C. (2013).** Recursive deep models for semantic compositionality over a sentiment treebank. *Proceedings of the 2013 Conference on Empirical Methods in Natural Language*, pp. 1631–1642.
9. **Lai, S., Xu, L., Liu, K., & Zhao, J. (2015).** Recurrent convolutional neural networks for text classification. *Twenty-ninth AAAI Conference on Artificial Intelligence*, pp. 2267–2273.
10. **Zhou, C., Sun, C., Liu, Z., & Lau, F. (2015).** A C-LSTM Neural Network for Text Classification. arXiv preprint arXiv:1511.08630, pp. 1–10.
11. **Cer, D., Yang, Y., Kong, S.Y., Hua, N., Limtiaco, N., John, R.S., & Sung, Y.H. (2018).** *Universal sentence encoder*. arXiv preprint arXiv:1803.11175, pp. 1–7.
12. **Pang, B. & Lee, L. (2005).** Seeing stars: Exploiting class relationships for sentiment categorization with respect to rating scales. *Proceedings of the 43rd annual meeting on association for computational linguistics, Association for Computational Linguistics*, pp. 115–124. DOI: 10.3115/1219840.1219855.
13. **Li, X. & Roth, D. (2002).** Learning question classifiers. *Proceedings of the 19th International Conference on Computational Linguistics Association for Computational Linguistics*, Vol. 1, pp. 1–7. DOI: 10.3115/1072228.1072378.
14. **Zhang, X., Zhao, J., & LeCun, Y. (2015).** Character-level convolutional networks for text classification. *Advances in Neural Information Processing Systems*, pp. 649–657.
15. **Pang, B. & Lee, L. (2004).** A sentimental education: Sentiment analysis using subjectivity summarization based on minimum cuts. *Proceedings of the 42nd Annual Meeting on Association for Computational Linguistics*. DOI: 10.3115/1218955.1218990.  
**Wang, J., Wang, Z., Zhang, D., & Yan, J. (2017).** Combining Knowledge with Deep Convolutional Neural Networks for Short Text Classification. *IJCAI*, pp. 2915–2921.

Article received on 25/01/2019; accepted on 04/03/2019.  
Corresponding author is Ajay Nagar.

# The Role of WordNet Similarity in the Affective Analysis Pipeline

Alejandra Segura Navarrete<sup>1</sup>, Christian Vidal-Castro<sup>1</sup>,  
Clemente Rubio-Manzano<sup>1</sup>, Claudia Martínez-Araneda<sup>2</sup>

<sup>1</sup> Universidad del Bío-Bío,  
Facultad de Ciencias Empresariales, Concepción,  
Chile

<sup>2</sup> Universidad Católica de la Santísima Concepción,  
Facultad de Ingeniería, Concepción,  
Chile

{asegura,clrubio,cvidal}@ubiobio.cl, cmartinez@ucsc.cl

**Abstract.** Sentiment Analysis (SA) is a useful and important discipline in Computer Science, as it allows having a knowledge base about the opinions of people regarding a topic. This knowledge is used to improve decision-making processes. One approach to achieve this is based on the use of lexical knowledge structures. In particular, our aim is to enrich an affective lexicon by the analysis of the similarity relationship between words. The hypothesis of this work states that the similarities of the words belonging to an affective category, with respect to any other word, behave in a homogeneous way within each affective category. The experimental results show that words of a same affective category have a homogeneous similarity with an antonym, and that the similarities of these words with any of their antonyms have a low variability. The novelty of this paper is that it builds the bases of a mechanism that allows incorporating the intensity in an affective lexicon automatically.

**Keywords.** Natural language processing, computational linguistics, affective computing, sentiment analysis, knowledge representation.

## 1 Introduction

Nowadays, Sentiment Analysis is a useful and important discipline in Computer Science, which allows obtaining potentially valuable knowledge about user's perceptions, expectations and attitudes in order to improve the decision-making process regarding products and marketing

strategies, among other uses. Sentiment Analysis has not only been applied in business but also in very different areas such as recommender systems [1, 2], electoral analysis [3] management of virtual museums [4], multilingual processing [5, 6] among others.

In general terms, there are two approaches to perform this type of analysis according to [7]. The first one uses a corpus of tagged texts that allows the construction of a classifier trained to execute this task. This approach uses supervised learning techniques that come from machine learning and statistics [8]. The second approach uses lexical resources, such as dictionaries or lexicons which are defined as a previously tagged set of words [9], i.e., every word is tagged according to its orientation [10]. There are two main lines of work: The identification of both positive and negative opinions, emotions and evaluations, using computing tools to assign a polarity to the content [11].

The estimation of the affective aspect of a text [9] called Affective Analysis, where there is a lexicon containing a set of words classified according to the emotions they represent [12, 13]. The emotion expressed in a sentence or text is obtained considering the emotion of all the words contained in that text [14].

The classification process in Sentiment Analysis is simpler than in Affective Analysis.

The first one classifies in two or three categories, either {positive, negative} or {positive, negative or neutral} and the second one can do so in many, depending on the model of emotions used. In addition, a text can express more than one emotion and even two different texts can express the same emotion but with different intensities.

In Affective Analysis based on lexicon approach, the results depend on the quality and completeness of the lexicon used in the process. The affective lexicons include the words grouped by affective category. This fact only allows a words-bag analysis since the words of each affective category do not contain affective intensity information; therefore, it is not possible determining affective profiling of a document. In Sentiment Analysis, there are works having improved the quality of affective lexicons by adding information such as valence, arousal and dominance [15–17]. In the case of affective analysis based on lexicon, studies are mainly aimed at increasing the number of words of an affective lexicon [18].

The objective of this paper is to enrich a lexicon of affects by the analysis of the similarity relationship between words. The hypothesis of this work states that the similarities of the words belonging to an affective category, with respect to any other word, behave in a homogeneous way within each affective category. We found evidences that similarities of the words belonging to an affective category, with respect to any other word, behave in a homogeneous way within each affective class. This finding will allow us to determine intensities for the emotions of an affective category and to improve automatic enrichment process of affective lexicons.

The rest of the article is structured as follows: Chapter 2 presents a background and a brief state of the art about the use of lexicons in affective analysis and similarity measures between two words. Chapter 3 presents the hypothesis and experiments performed to prove it. Chapter 4 presents the results and discusses the word's similarity behavior of the lexicon's affective classes. Finally, conclusions and future work lines are presented in Chapter 5.

## 2 Background and Related Work

In lexicons commonly used in Affective Analysis, consider different classifications of basic emotions, assuming that all other emotions would depend on these subsets.

For example, author proposed 6 categories in [19]: anger, disgust, fear, joy, sadness and surprise. In [20] was proposed an affective lexicon called WordNet-Affect, which was built based on the WordNet knowledge base, through the selection and tagging of affective concepts. This initial base was extended using sentences and patterns extracted from Open Mind Commonsense [21]. WordNet-Affect classifies words into the six categories of Ekman.

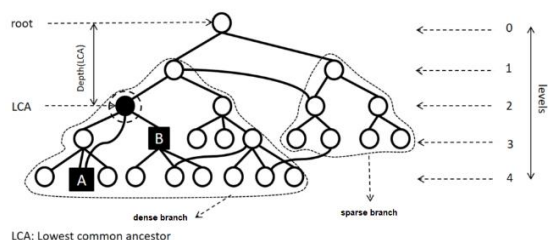
Each word in the lexicon contains lexical and affective information, for example, the role of the word in speech (part-of-speech), classification according to emotion theory or representation, among others. Another affective lexicon was proposed in [22], which considered 8 affective categories: anger, anticipation, disgust, fear, joy, sadness, surprise and trust. This lexicon was generated from a list of affective words extracted from the Thesaurus WordNet-Affect and the most frequent words in Google n-gram Corpus [23].

In the case of Affective Analysis based on lexicon, the studies have as main objective to increase the number of words of an affective lexicon. For example, the authors in [18] present an approach for the Japanese language where a similarity metric was used to expand a small group of emotionally-charged words (containing 503 nouns) into an emotions dictionary (containing 15612 verbs). Other studies have improved the lexical resources through the integration [24] or the creation of lexicons for specific domains [25, 26].

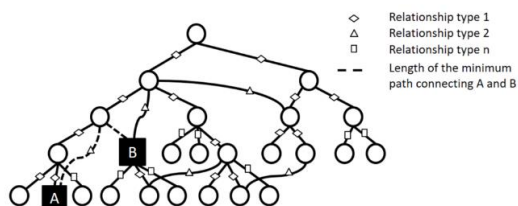
One aspect to consider is that a lexicon can be used in different domains. This may imply that one word may represent different affects in different domains [27]. On the other hand, in order to incorporate the concept of semantic similarity among words in Affective Analysis, it is necessary to analyze both, the metrics based on the structure and the metrics based on the Information Content (IC).

The semantics' similarity measures based on structure add variables such as lowest common





**Fig. 1.** Depth of the lowest common predecessor



**Fig. 2.** Local density of the subtree, distance between concepts and types of relationships involved between two concepts

ancestor (LCA) or least common subsumer (LCS), local specificity of the subtree that contains the concepts, the distance between concepts and the types of relationships involved between them (as shown in Figure 1 and Figure 2).

For example, in [28] the Path measure was proposed, which is based on the shortest path that connects the senses in the “is-a” relationship of WordNet (1). Wu & Palmer measure, proposed in [29] is calculated based on the depth of the LCA and the number of links between concepts and predecessor (2). The measure proposed in [30] is calculated based on the depth of each of the concepts, the LCA depth and the shortest distance between concepts. Another proposal from the same authors [31] also includes local specificity. Finally, in [32] the shortest route between concepts, LCA depth and empiric information are considered.

In addition to the previous ones, the Leacock & Chodorow metric (3) was used in this work, since it determines how similar two senses are, based on the shortest path that connects the senses (as above) and the maximum depth of them:

$$Sim_{Path}(c1, c2) = -\log(pathlen(c1, c2)), \quad (1)$$

$$Sim_{Wu\&P}(c1, c2) = \frac{2 \cdot Depth(LCA)}{(Depth1 + Depth2)}, \quad (2)$$

$$Sim_{L\&CH}(c1, c2) = -\log \frac{Len(LCA)}{(2 \cdot maxDepth(LCA.pos))}. \quad (3)$$

where  $Depth1 = \min(depth(\{tree\ in\ T1 | tree\ contains\ LCA\}))$   
and  $Depth2 = \min(depth(\{tree\ in\ T2 | tree\ contains\ LCA\}))$

Semantics' similarity measures based on information content consider the IC of the nodes derived from the model and statistical corpus, for example, through measures such as term frequency and inverse document frequency (tf-idf). The more information (IC) they share; the more similar concepts are. Some measures in this category are the ones proposed in [32-34]. In Resnik's metric [32], the IC of the LCA is considered, and Jiang & Conrath's and Lin's proposals [33-34] include improvements to Resnik's measure as shown in (4), where the IC of each of the concepts is added. Lin's measure (5) is based on Jiang & Conrath's proposal (6):

$$Sim_{Res}(c1, c2) = -\log P(LCA(c1, c2)), \quad (4)$$

$$Sim_{Lin}(c1, c2) = \frac{2 \cdot IC(LCA)}{IC(c1) + IC(c2)}, \quad (5)$$

$$Sim_{J\&C}(c1, c2) = \frac{1}{(IC(c1) + IC(c2) - 2 \cdot IC(LCA))}, \quad (6)$$

re  $c1$  and  $c2$  are the concepts compared

$LCA(c1, c2)$  is lowest common ancestor between  $c1$  y  $c2$

$P(c)$  is the probability of encountering an instance of concept  $c$

$IC(LCA)$  is an information content of LCA

$IC(c1)$  is an information content of  $c1$

$IC(c2)$  is an information content of  $c2$

Recent works incorporate a metric [35] that uses IC and the amount of nodes to try to simplify and improve the calculation of similarity between pairs of words contained in graphs. In sentiment analysis based on lexical approach, there are works having improved the quality of affective lexicons by adding new information such as valence, arousal and dominance [15-17].

The above, basing on pointwise mutual information (PMI), latent semantic analysis (LSA)

and the semantic proximity determine by a co-occurrence between the words and the benchmarks to obtain an index of proximity.

### 3 Method

To prove our hypothesis an experiment was designed using the affective lexicon in English proposed in [20], due to its availability and reputation in affective computing area. Such a lexicon has 1080 words ( $w_1, w_2 \dots w_{1080}$ ), grouped into six affective categories (including repetitions). Anger category has 242 words, disgust has 50, fear has 143, joy has 387, sadness has 195 and surprise has 63 words.

From each of the categories the words selected were those with an affective connotation, some 371 in total (34.35%). The concept of affective connotation is used to refer to words that have at least a synset in WordNet, whose hypernyms coincides with at least one of the following concepts emotion, affect, emotionality, feeling and moods, which hereinafter will be called affective ancestor. From the words selected, it was detected that four belong to more than one affective category; these are  $\text{horror} \in \{\text{fear}, \text{disgust}\}$ ,  $\text{dismay} \in \{\text{fear}, \text{sadness}\}$ ,  $\text{suspense} \in \{\text{joy}, \text{fear}\}$  and  $\text{admiration} \in \{\text{joy}, \text{surprise}\}$ . Out of the 371 words selected, 100 belong to the anger category, 6 to disgust, 46 belong to fear, 145 to joy, 66 to sadness and 8 to surprise.

Considering that similarity between two words indicates the closeness between them, this work calculated the similarity between words of an affective category and an antonym.

The experiment was divided into two parts: The first one analyzes the behavior of words of an affective category based on the similarity of these words with an antonym (as shown in Figure 3). Regarding this, it is expected that, within each affective category, the variability of the similarity of words with their antonym is homogeneous. The second one (as shown in Figure 4) analyzes the similarity of words with 3 antonyms of the affective category. For this case, it is expected that the variability of the similarity of words, with each of their antonyms, is homogeneous and low.

The opinion of an expert in English language was used to identify antonyms, who selected the three best antonyms for each affective category.

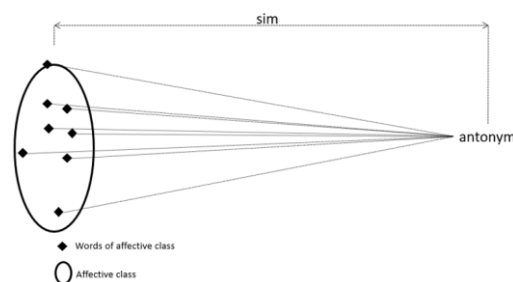


Fig. 3. Research hypothesis and experiments (1/2)

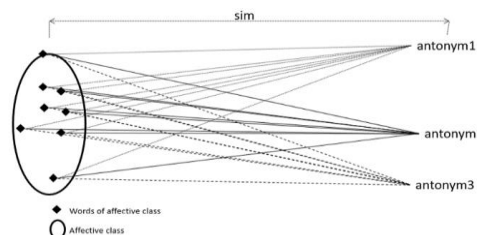
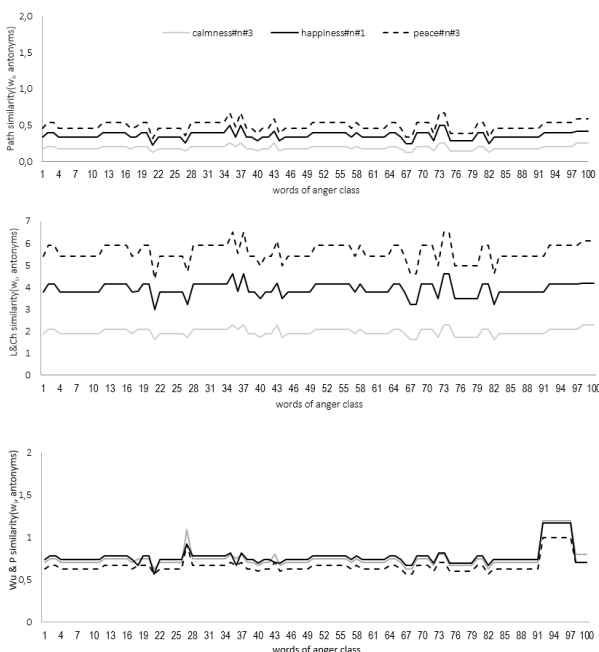


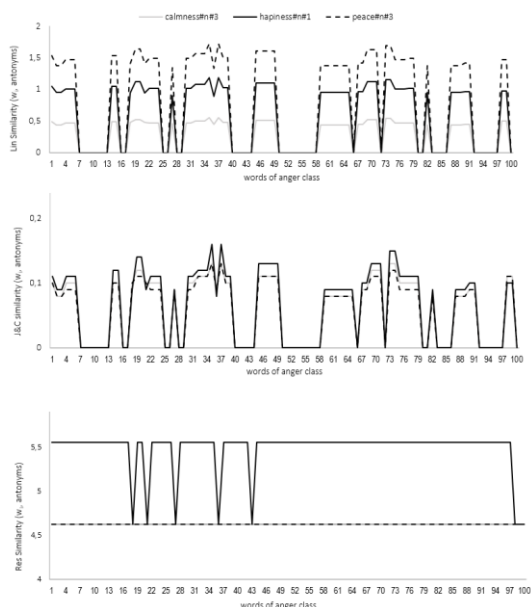
Fig. 4. Research hypothesis and experiments (2/2)

Table 1. List of antonyms for affective class

Class	Ranking	Antonym
anger	1	happiness#n#1
	2	calmness#n#3
	3	peace#n#3
disgust	1	fondness#n#1
	2	admiration#n#1
	3	love#n#1
fear	1	fearlessness#n#1
	2	bravery#n#2
	3	confidence#n#2
joy	1	sorrow#n#1
	2	sadness#n#1
	3	melancholy
sadness	1	happiness#n#1
	2	joy#n#1
	3	gladness#n#1
surprise	1	expectation#n#3
	2	calmness#n#3
	3	coolness#n#2



**Fig. 5.** Behavior of metrics based on structure applied to words of the anger affective category, with their three antonyms



**Fig. 6.** Behavior of the three metrics based on IC applied to words of the anger affective category, with their three antonyms

It is worth mentioning that, although WordNet does not provide an antonym for each affective category, the ones proposed by experts are available in WordNet.

Table 1 shows the three antonyms arranged according to the importance determined by the expert. It was verified that each antonym also had an affective connotation, in the table the antonym is represented including the word itself, the part of the speech and the meaning given by the affective connotation (word#pos#sensenumber).

To obtain similarities between the words of each affective category and each antonym, a Java application was developed which uses WS4J<sup>1</sup> (WordNet Similarity for Java), API for several Semantic Relatedness/Similarity algorithms.

## 4 Results and Discussion

The behavior analysis of the similarity between words, and their three antonyms using the six metrics allows observing a homogeneous behavior in each category, this is, the ranges of values in which the similarities move are small. As an example, Figure 5 and Figure 6 show the results of the six-metrics obtained for the words of the anger category considering their three antonyms (calmness#n#3, happiness#n#1, peace#n#3).

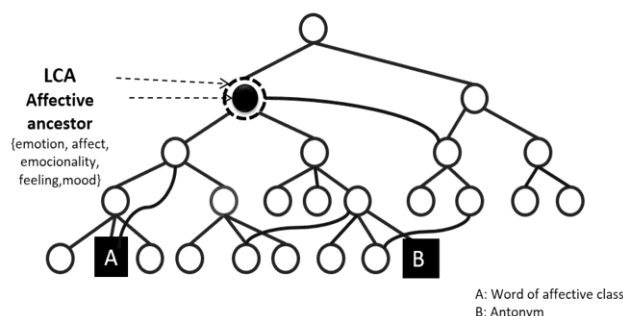
In the Resnik's case, the similarity provides modest results for all words. This can be explained by the fact that his calculation is based exclusively in IC of the LCA (4), where the closest common ancestor is determined, which for many words of the category and their antonyms, is the same, and coincides with the words used to differentiate the affective character of a word (as shown in Figure 7).

For example, considering the word gladness#n#1 as antonym of the sadness category and the word  $w_1$ =dispiritedness#n#1 as part of it, the result of Resnik's metric is  $Sim_{Res}(antonym, w_1) = 4.627$ , a value that repeats for 100% of the sadness category and 98% of all the words of the other affective categories, since most words share the common ancestor feeling#n#1.

<sup>1</sup> <http://ws4jdemo.appspot.com>

**Table 2.** Variabilities of the similarity of words with their antonyms per category

2	Antonym	Wu&P	J&C	L&CH	LIN	RES	PATH
anger	calmness#n#3	0.123	0.051	0.159	0.237	0.000	0.029
	happiness#n#1	0.108	0.057	0.159	0.271	0.252	0.028
	peace#n#3	0.093	0.049	0.117	0.234	0.000	0.017
disgust	admiration#n#1	0.015	0.037	0.067	0.180	0.000	0.011
	fondness#n#1	0.015	0.034	0.067	0.169	0.000	0.011
	love#n#1	0.041	0.050	0.151	0.210	0.347	0.030
fear	bravery#n#2	0.037	0.039	0.160	0.185	0.000	0.031
	confidence#n#2	0.032	0.038	0.117	0.184	0.000	0.018
	fearlessness#n#1	0.037	0.039	0.160	0.185	0.000	0.031
joy		0.086	0.044	0.190	0.218	0.576	0.031
	melancholy#n#1						
	sadness#n#1	0.096	0.053	0.225	0.238	0.576	0.044
	sorrow#n#1	0.086	0.045	0.190	0.220	0.576	0.031
sadness		0.081	0.000	0.145	0.000	0.000	0.026
	gladness#n#1						
	joy#n#1	0.080	0.054	0.139	0.244	0.159	0.025
	happiness#n#1	0.074	0.052	0.139	0.231	0.614	0.025
surprise	calmness#n#3	0.034	0.013	0.158	0.024	0.000	0.042
	coolness#n#2	0.018	0.004	0.058	0.005	0.000	0.006
	expectation#n#3	0.034	0.013	0.158	0.027	0.000	0.042



**Fig. 7.** Low similarity case in Resnik

The IC calculation proposed in Resnik (5) is based on the frequency words in corpus, for this predecessor's example feeling#n#1 is 4.627.

In general terms, for the metrics based on the IC, a low similarity or a similarity equal to zero could be due to the low frequency of some of the concepts in the vocabulary, even when both concepts are semantically related.

This could explain many of the values equal to zero obtained with the Jiang & Conrath and Lin metrics. In most cases, these values repeat for the same words regarding their three antonyms.

In addition to this, when Lin's metric is undetermined (5), the implementation of the API WS4J results in a zero value.

**Table 3.** Variabilities of similarities between antonyms

Class		Wu&P	J&C	L&CH	LIN	RES	PATH
anger	max	0.088065632	0.014142136	0.188561808	0.044969125	0.4384062	0.0377124
	min	0.028284271	0	0.103708995	0	0	0.0094281
	max-min	0.059781361	0.014142136	0.084852814	0.044969125	0.4384062	0.0282843
disgust	max	0.051854497	0.030912062	0.188561808	0.078740079	0.4384062	0.0377124
	min	0	0	0	0	0	0
	max-min	0.051854497	0.030912062	0.188561808	0.078740079	0.4384062	0.0377124
fear	max	0.042426407	0.004714045	0.188561808	0.004714045	0	0.0377124
	min	0.032998316	0	0.11785113	0	0	0.0141421
	max-min	0.00942809	0.004714045	0.070710678	0.004714045	0	0.0235702
joy	max	0.051854497	0.014142136	0.136707311	0.026246693	1.11E-16	0.0377124
	min	0.004714045	0	0.032998316	0	0	0
	max-min	0.047140452	0.014142136	0.103708995	0.026246693	1.11E-16	0.0377124
sadness	max	0.070395707	0.098432154	0.230362034	0.34127213	1.5167143	0.0449691
	min	0.018856181	0	0.061282588	0	0	0.0094281
	max-min	0.051539526	0.098432154	0.169079446	0.34127213	1.5167143	0.035541
surprise	max	0.188561808	0.035590261	0.565685425	0.224251843	1.8149074	0.108423
	min	0.164991582	0.023570226	0.414835978	0.193448242	1.8149074	0.0565685
	max-min	0.023570226	0.012020035	0.150849447	0.030803601	0	0.0518545

The same API yields a zero value when the IC of any of the two words is zero. The variabilities between the similarities of the words of each category with their antonyms, this is  $[\text{Sim}(\text{antonym}_1, w_1), \text{Sim}(\text{antonym}_1, w_2), \text{Sim}(\text{antonym}_1, w_n)]$  are summarized in Table 2.

The minimum standard deviation value is zero and the maximum value is 0.614. Regarding minimum values, these were obtained in the following cases: for the 3 antonyms of the surprise category, for the 3 antonyms of the fear category, for 2 antonyms of disgust, for 2 antonyms of anger and for 1 antonym of sadness.

These variability values were mainly obtained in Resnik's metric. However, minimum variability was also obtained with Jiang & Conrath and Lin's

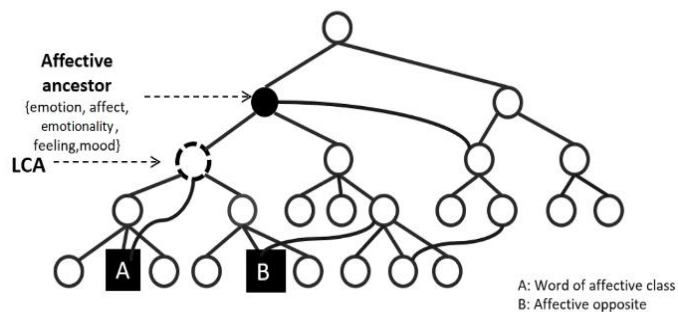
metric for the antonym *gladness#n#1* of the *sadness* category. On the other hand, the maximum variability value was obtained for the antonym *happiness#n#1* of the *sadness* category with Resnik's metric.

In general terms, as is presented in Table 2, the deviations obtained were low. It is worth mentioning that a certain degree of variability in the results is reasonable since in a same category there are words with different affective intensity.

For example, when calculating similarities of the words *grief#n#1* and *sorrow#n#1*, both belonging to the *sadness* category, with their antonym *joy#n#1*,  $\text{Sim}_{\text{Lin}}(\text{joy#n#1}, \text{grief#n#1}) = 0.50$  and  $\text{Sim}_{\text{Lin}}(\text{joy#n#1}, \text{sorrow#n#1}) = 0.52$  and the  $\text{Sim}_{\text{Wu\&P}}(\text{joy#n#1}, \text{grief#n#1}) = 0.70$  and

**Table 4.** Similarity between antonyms for each affective category and each metric

Class Similarity between		Wu&P	J&C	L&CH	LIN	RES	PATH
anger	calmness-happiness	0.75	0.9	2.07	0.474	4.62	0.2
	happiness-peace	0.66	0.09	1.74	0.4664	4.62	0.14
	peace-calmness	0.87	3.14	2.59	0.9833	9.36	0.33
	$\sigma$	0.02	4.98	0.36	0.1755	14.98	0.02
disgust	admiration-fondness	0.87	0.25	2.59	0.806	8.15	0.33
	fondness-love	0.75	0.10	2.07	0.4981	4.62	0.2
	admiration-love	0.75	0.12	2.07	0.5294	4.62	0.2
	$\sigma$	0.01	0.012	0.17	0.0574	8.29	0.01
fear	fearlessness-bravery	1	1.3E+07	3.68	1	9.81	1
	confidence-fearlessness	0.87	6.48	2.59	0.9922	9.82	0.33
	bravery-confidence	0.87	6.48	2.59	0.9922	9.82	0.33
	$\sigma$	0.01	1.1E+14	0.80	0.00004	0.00004	0.3
joy	melancholy-sadness	0.93	0.56	2.99	0.903	8.21	0.5
	sadness-sorrow	0.93	0.67	2.99	0.9175	8.21	0.5
	melancholy-sorrow	0.87	0.30	2.59	0.8352	8.21	0.33
	$\sigma$	0.002	0.071	0.10	0.0039	0	0.02
sadness	gladness-joy	0.75	0	2.07	0	4.62	0.2
	happiness-joy	0.82	0.14	2.30	0.613	5.55	0.25
	gladness-happiness	0.70	0	1.89	0	4.62	0.17
	$\sigma$	0.007	0.013	0.08	0.2505	0.57	0
surprise	calmness-coolness	0.5	0.06	1.49	0.227	2.39	0.11
	coolness-expectation	0.5	0.06	1.49	0.237	2.39	0.11
	calmness-expectation	0.85	0.11	2.59	0.5189	4.62	0.33
	$\sigma$	0.08	0.001	0.80	0.0549	3.31	0.03

**Fig. 8.** Use of opposite concept to improve problem of common affective ancestor

$\text{Sim}_{\text{Wu\&P}}(\text{joy}\#n\#1, \text{sorrow}\#n\#1) = 0.75$   
 were obtained.

This would show there are variations of  
 similarity between words of a same category,

indicating differences in the affective intensity of words. On the other hand, the standard deviation of the similarity of each word with their 3 antonyms was calculated, and since the antonyms were selected by the expert as the best antonyms, it is logical to expect a similar behavior of the words of a same category, regarding similarity, regardless of the antonym chosen.

In general terms, it is expected that the similarity calculation of a same word with the three antonyms yields a low variability. This is ratified in all affective categories; there were even cases with zero deviation.

As shown in Table 3, the similarities calculated between the words of the category and their antonyms, in most of the metrics, have a low variability, which leads us to the verification of the hypothesis. Results shown in Table 4 indicate that the antonym's selection performed by the experts was accurate since antonyms are similar to each other. As shown in Table 3, for each metric there were small differences between minimums and maximums, especially low in the fear category for each of the metrics.

During this work, the use of antonyms to analyze the behavior of the words of each affective category was very useful, since it allowed validating that similarity has a homogeneous behavior and a low variability. It is important to highlight that in order to analyze the similarity values and their relationship with the affective intensity or the diffuse belonging of a word to more than one affective category, it is necessary to use another word as a pivot, not an antonym, just an opposite word. This way, the problem represented in Figure 7 would be eliminated, and more significant values of similarity would be obtained for future analysis as visualized in Figure 8.

## 5 Conclusions and Future Work

This article evidences that words of a same affective category have a homogeneous similarity, as stated in the hypothesis. This statement is supported by the results, which show a low standard deviation of the similarity of words that make up an affective category.

The results obtained so far show the usefulness of similarity to enrich a lexicon, for example, when identifying words regarding their

diffuse classification or when determining the intensity of words that belong to a same affective category. Regarding this, it is possible to add words that do not have affective ancestors to a lexicon tagged with intensities using synonymy relationships. For this, it is important to identify the words that will be used as pivots. Although an antonym was used in this work, we believe that an intensity analysis requires a pivot that provides more meaningful information and that reduces the problem of metric's calculation based on IC, explained in the previous section. A priori, we believe it would be possible to obtain better results if a word with an opposite affective sense is used. For example, using Plutchik's taxonomy (8 emotions), this would require the re-classification of the lexicon based on Ekman's taxonomy (6 emotions).

The existence of mechanisms that improve the treatment of antonym relationships in WordNet, as well as the implementation of other similarity semantic metrics, would allow, with less effort, to improve affective knowledge bases, such as lexicons or dictionaries, considering, for example, the use of relationships of transitivity and of the semantic type for the analysis of knowledge structures.

## Acknowledgements

This paper is the result of work by the SOMOS research group (Software – MOdelling – Science), funded by the Dirección de Investigación and Facultad de Ciencias Empresariales of the Universidad del Bío-Bío, Chile. The authors thank to Facultad de Ingeniería de la Universidad Católica de la Santísima Concepción, Chile.

## References

1. Sun, J., Wang, G., Cheng, X., & Fu, Y. (2015). Mining affective text to improve social media item recommendation. *Information Processing & Management*, Vol. 51, No. 4, pp. 444–457. DOI: 10.1016/j.ipm.2014.09.002.
2. Gurini, D.F., Gasparetti, F., Micarelli, A., & Sansonetti, G. (2018). Temporal people-to-people recommendation on social networks with sentiment-based matrix factorization. *Future*

- Generation Computer Systems*, Vol. 78, No. 1, pp. 430–439. DOI: 10.1016/j.future.2017.03.020.
3. **Mohammad, S.M., Zhu, X., Kiritchenko, S., & Martin, J. (2015).** Sentiment, emotion, purpose, and style in electoral tweets. *Information Processing & Management*, Vol. 51, No. 4, pp. 480–499. DOI: 10.1016/j.ipm.2014.09.003.
  4. **Bertola, F. & Patti, V. (2016).** Ontology-based affective models to organize artworks in the social semantic web. *Information Processing & Management*, Vol. 52, No. 1, pp. 139–162. DOI: 10.1016/j.ipm.2015.10.003.
  5. **Balahur, A. & Perea-Ortega, J.M. (2015).** Sentiment analysis system adaptation for multilingual processing: The case of tweets. *Information Processing & Management*, Vol. 51, No. 4, pp. 547–556. DOI: 10.1016/j.ipm.2014.10.004.
  6. **Severyn, A., Moschitti, A., Uryupina, O., Plank, B., & Filippova, K. (2016).** Multi-lingual opinion mining on YouTube. *Information Processing & Management*, Vol. 52, No. 1, pp. 46–60. DOI: 10.1016/j.ipm.2015.03.002.
  7. **Taboada, M., Brooke, J., Tofiloski, M., Voll, K., & Stede, M. (2011).** Lexicon-based methods for sentiment analysis. *Computational linguistics*, Vol. 37, No. 2, pp. 267–307. DOI: 10.1162/COLI\_a\_00049.
  8. **Pang, B., Lee, L., & Vaithyanathan, S. (2002).** Thumbs up?: sentiment classification using machine learning techniques. *Proceedings of the ACL-02 conference on Empirical methods in natural language processing*, Vol. 10, pp. 79–86. DOI: 10.3115/1118693.1118704.
  9. **Wilson, T., Wiebe, J., & Hoffmann, P. (2005).** Recognizing contextual polarity in phrase-level sentiment analysis. *Proceedings of Human Language Technology Conference and Conference on Empirical Methods in Natural Language Processing*, pp. 347–354.
  10. **Carrillo de Albornoz, J. (2011).** *Un modelo lingüístico-semántico basado en emociones para la clasificación de textos según su polaridad e intensidad.*
  11. **Grefenstette, G., Qu, Y., Shanahan, J.G., & Evans, D.A. (2004).** Coupling niche browsers and affect analysis for an opinion mining application. *Proceedings of Recherche d'Information Assistée par Ordinateur (RIAIO).*
  12. **Ekman, P. (1993).** Facial expression and emotion. *American psychologist*, Vol. 48, No. 4, pp. 384.
  13. **Strapparava, C. & Valitutti, A. (2004).** Wordnet affect: an affective extension of wordnet. *Lrec*, pp. 1083–1086.
  14. **D'Urso, V. & Trentin, R. (2007).** *Introduzione alla psicologia delle emozioni.* Laterza.
  15. **Bestgen, Y. (2002).** Détermination de la valence affective de termes dans de grands corpus de textes. *Actes du Colloque International sur la Fouille de Texte*, pp. 1–14.
  16. **Redondo, J., Fraga, I., Padrón, I., & Comesaña, M. (2007).** The Spanish adaptation of ANEW (affective norms for English words). *Behavior research methods*, Vol. 39, No. 3, pp. 600–605. DOI: 10.3758/BF03193031.
  17. **Turney, P.D. & Littman, M.L. (2003).** Measuring praise and criticism: Inference of semantic orientation from association. *ACM Transactions on Information Systems (TOIS)*, Vol. 21, No. 4, pp. 315–346. DOI: 10.1145/944012.944013.
  18. **Ptaszynski, M., Rzepka, R., Araki, K., & Momouchi, Y. (2014).** Automatically annotating a five-billion-word corpus of Japanese blogs for sentiment and affect analysis. *Computer Speech & Language*, Vol. 28, No. 1, pp. 38–55. DOI: 10.1016/j.csl.2013.04.010.
  19. **Ekman, P. (1973).** Cross-cultural studies of facial expression. *Darwin and facial expression: A century of research in review*, pp.169–222.
  20. **Valitutti, A., Strapparava, C., & Stock, O. (2004).** Developing affective lexical resources. *PsychNology Journal*, Vol. 2, No. 1, pp. 61–83.
  21. **Singh, P., Lin, T., Mueller, E.T., Lim, G., Perkins, T., & Zhu, W.L. (2002).** Open Mind Common Sense: Knowledge acquisition from the general public. *OTM Confederated International Conferences on the Move to Meaningful Internet Systems*, pp. 1223–1237. DOI: 10.1007/3-540-36124-3\_77.
  22. **Mohammad, S.M. & Turney, P.D. (2013).** Crowdsourcing a word–emotion association lexicon. *Computational Intelligence*, Vol. 29, No. 3, pp. 436–465. DOI: 10.1111/j.1467-8640.2012.00460.x.
  23. **Brants, T. & Franz, A. (2006).** *Web 1T 5-gram Version 1 LDC2006T13.*
  24. **Cho, H., Kim, S., Lee, J., & Lee, J.S. (2014).** Data-driven integration of multiple sentiment dictionaries for lexicon-based sentiment classification of product reviews. *Knowledge-Based Systems*, Vol. 71, pp. 61–71. DOI: 10.1016/j.knosys.2014.06.001.
  25. **Huang, S., Niu, Z., & Shi, C. (2014).** Automatic construction of domain-specific sentiment lexicon based on constrained label propagation. *Knowledge-Based Systems*, Vol. 56, pp. 191–200. DOI: 10.1016/j.knosys.2013.11.009.
  26. **Park, S., Lee, W., & Moon, I.C. (2015).** Efficient extraction of domain specific sentiment lexicon with



- active learning. *Pattern Recognition Letters*, Vol. 56, pp. 38–44. DOI: /10.1016/j.patrec.2015.01.004.
27. **Hung, C. (2017).** Word of mouth quality classification based on contextual sentiment lexicons. *Information Processing & Management*, Vol. 53, No. 4, pp. 751–763. DOI: 10.1016/j.ipm.2017.02.007.
  28. **Rada, R., Mili, H., Bicknell, E., & Blettner, M. (1989).** Development and application of a metric on semantic nets. *IEEE transactions on systems, man, and cybernetics*, Vol. 19, No. 1, pp. 17–30.
  29. **Wu, Z. & Palmer, M. (1994).** Verbs semantics and lexical selection. *Proceedings of the 32nd annual meeting on Association for Computational Linguistics*, pp. 133–138. DOI: 10.3115/ 981732.981751.
  30. **Al-Mubaid, H. & Nguyen, H.A. (2006).** A cluster-based approach for semantic similarity in the biomedical domain. *International IEEE Conference in Medicine and Biology Society*, pp. 2713–2717.
  31. **Nguyen, H.A. & Al-Mubaid, H. (2006).** New ontology-based semantic similarity measure for the biomedical domain. *International IEEE Conference on Granular Computing*, pp. 623–628. DOI: 10.1109/GRC.2006.1635880.
  32. **Li, Y., Bandar, Z.A., & McLean, D. (2003).** An approach for measuring semantic similarity between words using multiple information sources. *IEEE Transactions on knowledge and data engineering*, Vol. 15, No. 4, pp. 871–882. DOI: 10.1109/TKDE.2003.1209005.
  33. **Jiang, J.J. & Conrath, D.W. (1997).** Semantic similarity based on corpus statistics and lexical taxonomy. *International Conference Research on Computational Linguistics (ROCLING X)*.
  34. **Resnik, P. (1999).** Semantic similarity in a taxonomy: An information-based measure and its application to problems of ambiguity in natural language. *Journal of artificial intelligence research*, Vol. 11, pp. 95–130. DOI: 10.1613/jair.514.
  35. **Gao, J.B., Zhang, B.W., & Chen, X.H. (2015).** A WordNet-based semantic similarity measurement combining edge-counting and information content theory. *Engineering Applications of Artificial Intelligence*, Vol. 39, pp. 80–88. DOI: 10.1016/j.engappai.2014.11.009.

Article received on 17/01/2019; accepted on 04/03/2019.  
Corresponding author is Alejandra Segura Navarrete.



# Understanding Blogs through the Lens of Readers' Comments

Abhilasha Sancheti<sup>1</sup>, Natwar Modani<sup>1</sup>, Gautam Choudhary<sup>2</sup>, Chintia Priyadarshini<sup>3</sup>,  
Sai Sandeep Moparthy<sup>4</sup>

<sup>1</sup> Adobe Research, Big data Experience Lab,  
Bangalore, India

<sup>2</sup> Indian Institute of Technology, Department of Computer Science,  
Roorkee, India

<sup>3</sup> Indian Institute of Technology, Department of Mathematics,  
Kanpur, India

<sup>4</sup> Indian Institute of Technology, Department of Computer Science,  
Bombay, India

{sancheti, nmodani}@adobe.com, {gc.iitr, priyadarshinichintia, moparthy.saisandeep}@gmail.com

**Abstract.** In order to keep their audience engaged, authors need to make sure that the blogs or articles they write cater to the taste of their audience and are understood by them. With the rapid proliferation of online blogging websites, the participation of readers by expressing their opinions and reviews has also increased in the form of comments on the blogs. These comments are valuable source for the authors to understand how their audience are perceiving their blogs. We believe that associating comments to the specific part of the blog they refer to will help author in getting insights about parts of the blog which are being discussed and the questions or concerns that readers have about those parts. Moreover, categorizing these comments will further aid the author in imbibing the comments. In this work, we describe a method to associate comments to the specific parts of the blog and introduce a hierarchical way to categorize the comments as Suggestion, Agreement, Disagreement or Question.

**Keywords.** Classification, comments association, support vector machine, feature selection.

## 1 Introduction

Social blogging platforms allow authors to share information, their personal experiences and opin-

ions on a wide variety of topics. These platforms also allow readers to leave their comments on the blog. The readers might agree or disagree with the author, provide suggestions for improving the blog or might have some questions. The authors need to make sure that the blogs they write cater to the taste of their audience and are understood by them so that they remain engaged. The primary mechanism for an author to understand how her audience are reacting to the blogs is to understand the comments they write for the blog. For the popular blogs, the number of comments can be large. Hence, if the author has created many blogs, reading all the comments, understanding the commenter's reaction, and which parts of the blog (called blog segment) need attention is a tedious and time-consuming task. Understanding what parts of the blog the audience did not understand (and had questions about), what parts did they agree/disagree with, and where did they feel the need to make changes to the blog can enable the authors to write the blogs in a more engaging way in future or to improve the existing blogs. In absence of a tool support, the authors do not fully benefit from insightful comments made by the

readers to improve the content of their future blogs (or make changes to existing ones).

In this work, we present a method for automatically associating the comments to the part of the blog they refer to and classifying the comments as Suggestion, Agreement, Disagreement or Question. Additionally, we provide a visual representation of this information, so that authors can quickly understand the type and strength of user reactions generated by the various parts of their blogs. This analysis can also help a reader in determining what blogs (and which parts of these blogs) should they read depending on their interests and roles. In today's world, where information overload is a major challenge, such insights can help them become more productive.

In Section 2 we detail the prior explorations in the field of comment association and classification. In Section 4 we describe the methodology in detail and present the results in Section 5 followed by the conclusion.

## 2 Related Work

To the best of our knowledge, there is no existing work which talks about comment association and classification simultaneously. Moreover, there is no work which classifies comments in the set of categories that we are considering, though there are different works which have looked at these categories separately.

### 2.1 Classification

Classification of user comments is a well-researched area. People have come up with various classification schemes for YouTube video comments [17], product reviews [3], tweets [5], etc. Most of these schemes talk about the type of sentiment, emotion [16] or mood expressed in the comment. There are a few works on identification of spam, off-topic, obscene, toxic and abusive [2] comments made on online blogs or YouTube videos<sup>1</sup>.

<sup>1</sup>[https://pdfs.semanticscholar.org/65fb/992b712d75c6499d8649d53ad575bdef9e0e.pdf?\\_ga=2.181395107.541616974.1532947452-1106483369.1517137894](https://pdfs.semanticscholar.org/65fb/992b712d75c6499d8649d53ad575bdef9e0e.pdf?_ga=2.181395107.541616974.1532947452-1106483369.1517137894)

There are also a few works which focus on the semantics or content aspects of the short texts. There are prior explorations [21, 22] on advice mining from the web forums which introduce various linguistic features which can be used to identify advices.

We leverage these features to classify sentences as suggestion. [22] proposed a hidden Markov model for labelling sequential sentences as advice revealing or not and use syntactic, semantic and contextual features for their task. Our task is different from theirs since their task involves independent comments and not sequential sentences.

With respect to the agreement and disagreement categories, there have been previous work on recognizing disagreement in informal political arguments [1] and identifying agreement and disagreement in the social media dialogues [11]. Among these work, [1] show that use of contextual and dialogue features improve accuracies as compared to unigrams. Topic independent features improve the performance of agreement-disagreement classification over unigrams as demonstrated by [11].

Apart from this, stance detection in tweets is also similar to identifying agreement and disagreement in text with respect to a part of the blog. A set of structural, contextual, sentiment and label-based features for predicting stance towards a mentioned target are defined in [9]. However, above mentioned approaches will not directly work in our case because of the differences in domain and also the dataset under consideration.

While there has been significant work on classifying short text, some of which also address comment classification, we are not aware of any work which classifies the comments in multiple classes, and in particular, the classes we are taking into consideration (agreement, disagreement, questions and suggestions).

We hypothesize that comments belonging to these classes will provide constructive and valuable insights about the blog to its author and other readers.

## 2.2 Association of Comments

There have been some research in the area of associating comments to a part of the news story and aligning comments to the news topics. An unsupervised technique is proposed in [18] which takes cosine similarity of LDA, SS-PLSA and BOW features of both comment and the segment of the news article to align comments with the segments. [19] propose a supervised technique for the task of alignment and show that the structured learning approach performs better than the other unsupervised and binary classification approaches. Frameworks for aligning comments to news topics by automatically extracting topics from a given news article and its associated comments are described in work like [7, 6]. However, in this work we position the comment association task as a 'question-answering' task where comment is considered as a query and the different parts of the blogs as the answer.

## 3 Problem Definition

The problem that we are trying to solve can be stated as: How to help authors and readers in extracting useful insights from the comments on a blog to:

- Help authors in understanding the audience reaction and improve upon her writing in future
- Help readers in understanding the blog through comments, possibly to prioritize the reading

In this work, we aim to solve the following sub-problems which define the useful insights that we will be presenting to the authors/readers.

1. Understanding the scope of the comment with respect to the blog. By this we mean associating the comment to the segment (defined at the level of a paragraph) of the blog it is referring to.
2. Understanding the type of the comment. We are considering the following types:
  - **Agreement:** comments that support (parts of) the content of the blog.

- **Disagreement:** comments that contradict/disapprove of (parts of) content in the blog.
- **Suggestion:** comments that advise or suggest changes to the content or suggests some alternatives to what's present in the blog.
- **Question:** comments that capture queries or doubts about (parts of) the content in the blog.

3. Visual representation of the comments related information for better insights

We hypothesize that the classes under consideration are exhaustive (after the removal of irrelevant comments) because there could be two scenarios in which reader can comment (1) reader does not understand the blog, and (2) the reader understands the blog. In scenario (1) he/she will have doubts with respect to the content of the blog and thus his/her comments will belong to the 'Question' category. In case (2) he/she will either have some suggestions for improving the content or will agree/disagree with the content. There might also be some comments, which go off-topic or are general point of views like "I don't like traveling alone" but we are not considering them since they can always be preprocessed and filtered out.

## 4 Methodology

In this section, we describe the dataset, the features, and the techniques used for the classification and the comment association task.

### 4.1 Data Preparation

We are not aware of previous work which simultaneously tackles the tasks of comment classification and its association with the part of the blog. Therefore, we curated a dataset to suit our purpose with the help of human annotators.

We collected a total of 90 blogs from different online blogging websites and asked the annotator to write comments on the presented blog indicating which part of the blog it corresponds to and

the category (suggestion, question, agreement, disagreement) it belongs to.

We had a total of 271 comments with 95 in Question, 70 in Disagreement, 674 in Agreement and 39 in Suggestion category. Since the size of our corpus was small we used auxiliary datasets for the training the classifiers individually. We used Open Domain Suggestion Mining dataset [12] for the suggestion classification task. This dataset consists of tagged sentences from various domains such as electronics reviews, hotel reviews, customer service reviews, and travel forums. We used a subset of Internet Argument Corpus (IAC) [20] for the agreement-disagreement classification task. This dataset consists of pairs of the kind (quote, response), where quote is the base sentence and the response either agrees or disagrees with the quote. There was a huge class imbalance in this data and thus we downsample sentences with disagreement label to get a balanced dataset.

## 4.2 Classification

We build separate classifiers for Suggestion, Question and Agreement/Disagreement for the training purpose and propose a hierarchical approach to classify the given test comment into one of the possible categories. The categories under consideration are mutually exclusive in our case and thus each comment can belong to only one category.

### 4.2.1 Suggestion Classifier

We train a binary SVM classifier for classifying the comment as suggestion or non-suggestion. Our main contribution is in feature engineering. We consider the following set of features for this classification task and show the performance improvement results in the next section.

1. **Clue Words (Clue):** We curated our own list of clue words (such as suggest, recommend, advice, urge, request, etc.) which were selected by investigating the training data. A binary feature vector of dimension equal to the number of clue words is created where the value corresponding to each dimension

denotes the presence or absence of a clue word in the comment.

2. **Modal Verbs (MV):** [21] has shown that advice revealing sentences often expresses modality which are expressed using the modal verbs (such as can, could, might, should, would, etc.). We define a set of modal verbs and a binary feature corresponding to each of the modal verb, indicating the presence or absence of the modal verb in the comment.
3. **Imperative Mood Expressions (IME):** [21] found that sentences containing imperative mood expressions (such as ‘do not bring mobile phones’, ‘it is a good idea to add more experimental results’) result in the actions in certain ways. We also used this feature in order to characterize the suggestions. We used the same heuristic method as defined by [21] for finding value of this feature. The heuristic says that if the verb present in the comment is not preceded by a subject, then most likely the comment contains an imperative mood expression.
4. **Typed Dependencies (TypDep):** We leveraged this feature as defined by [22]. We considered only conjunct, clausal subject, and nominal subject relations, which are denoted by “conj”, “csubj”, and “nsubj”, respectively in the comment’s parse tree obtained using Stanford Dependency Parser [4].
5. **Informativeness Score (InfScore):** It is the summation of the tf-idf score of all the words in the comment:

$$\sum_{w_i \in C} TfIdf(w_i),$$

where  $C$  is the comment and  $w_i$  is the word in the comment.

The training was done using the auxiliary dataset mentioned in above subsection. We used radial basis function (rbf) kernel while training the SVM.

#### 4.2.2 Question Classifier

We consider identification of question as a binary classification task. We use Stanford parser [10] for obtaining the parse tree of the given sentence and check for the presence of either of the two tags, namely SBARQ and SQ in the tree.

1. SBARQ: Presence of this tag indicates the presence of a direct question introduced by a wh-word or a wh-phrase.
2. SQ: Presence of this tag indicates the presence of an inverted yes/no question, or main clause of a wh-question, following the wh-phrase in SBARQ.

We mark the given sentence as belonging to a Question category if either of these tags is present in the parse tree of the comment.

#### 4.2.3 Agree-Disagree Classifier

We train a binary SVM classifier in order to classify the sentences into agreement and disagreement category. Presence of agreement or disagreement depends upon the context and thus for this classification task we also consider the segment of the blog that the comment is associated with. Each of the features we experimented with is explained below.

1. LIWC [14]: These features are used by [1] for recognizing disagreement in political arguments. We hypothesize that use of linguistic features will help in identifying the agreement and disagreement. We calculate the LIWC features for both the comment and the part of the blog it belongs to.
2. Glove Embedding (Glove) [15]: This feature represents the semantics of the comment. The feature value is a 50-dimensional vector obtained by the summation of the 50 dimensional embeddings of all the words present in the comment.

3. N-grams: We curated our own list of  $n$ -grams after investigating the data. These  $n$ -grams characterizes the presence or absence of agreement in the comment. For each  $n$ -gram, a binary value is assigned depending upon the presence or absence of that  $n$ -gram in the comment.
4. Positive and Negative sentiment words (Pos-Neg): We leverage the positive and negative sentiment words curated by [8] for identifying the polarity of product reviews to classify the comment as agreement or disagreement. We consider the difference in the number of positive and negative sentiment words present in the comment as the feature.
5. Positive Sentiment words (Pos): This feature denotes the number of positive sentiment words present in the comment.
6. Negative Sentiment words (Neg): This feature denotes the number of negative sentiment words present in the comment.
7. Affin score (Affin) [13]: It gives a word polarity score between -5 to +5. The feature value is the summation of the affin score of all the words in the comment. The same feature is also calculated for the part of the blog that the comment belongs to.

The training was done using the auxiliary dataset mentioned in the previous subsection. We used radial basis function (rbf) kernel while training the SVM. The model performed best with just the N-gram features.

#### 4.2.4 Hierarchical Classifier

Given a comment, we propose a hierarchical approach to classify it into one of the classes we are taking into consideration. We use the individually trained classifier models as described in the previous section with the features as per the best performing models.

Figure 1 shows the workflow of this classifier. The comments first go through the Suggestion classifier which if predicted as a suggestion, are classified as suggestion and not passed through

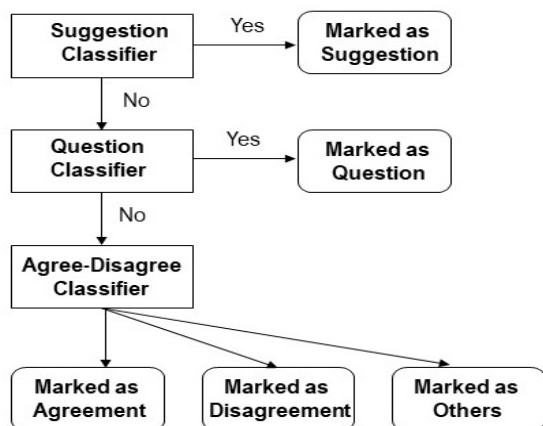


Fig. 1. Hierarchical Classification Workflow

any of the other classifiers. All the comments which are not classified as suggestions are fed to the Question classifier. If the comment is predicted as a question, then we label it as a question and pass the rest of the comments through the Agree-Disagree classifier.

This classifier finally classifies all the remaining comments into agreement or disagreement.

We believe that some suggestions like “Did you try deep Learning approaches?” are written in question form and will get mis-classified if they are first passed through the question classifier. Thus, we chose this hierarchy.

### 4.3 Association of Comments with Segment of the Blog

We model the association task as a ‘question-answering’ problem and use Learning to Rank models to rank different segments given a comment. We consider comment as the query and the segments of the blog as the answers.

Given a query the model ranks the answers. We used RankLib library’s ListNet<sup>2</sup> model for this purpose.

We use the following lexical and semantic features for our purpose:

<sup>2</sup><https://sourceforge.net/p/lemur/wiki/RankLib/>

### Lexical Features:

1. Segment Length (SegLen): Number of terms in the segment
2. Segment Position (SegPos): Relative position of the segment with respect to the blog
3. Exact Match (EMatch): It is a binary feature indicating whether the comment is a substring of the segment
4. Term Match (TMatch): Number of terms that are common in the comment and the segment
5. Synonym Match (SMatch): It is the fraction of comment’s terms whose synonym is present in the segment
6. Language Model(LM): It is a score which is computed as the log likelihood of the comment being generated from the segment

### Semantic Features:

7. Word2Vec Similarity (W2V): It is the cosine similarity score between the summation of the word2vec embeddings of the words in the comment and the segment
8. Universal Sentence Embedding (USE) Similarity: It is the cosine similarity score between the USE embeddings of the comment and the segment

## 5 Evaluation and Results

We evaluated comment association task on the following two metrics.

1. Mean Reciprocal Rank (MRR): It is given by:

$$\frac{1}{|C|} \sum_{i=1}^{|C|} \frac{1}{rank_i},$$

where C is the set of the comments that are queried for association and  $rank_i$  refers to the rank position of the correctly associated segment.



2. Percentage accuracy: It is the ratio of comments correctly associated to the total number of comments queried for the association.

Table 1 shows the results of the task of associating the comments with the segment of the blog. It can be inferred from the results that use of Word2Vec embeddings (W2V) is bringing down both MRR and accuracy values as compared to the case when it is not used. However, use of Universal Sentence Encoding (USE) feature along with all the other lexical features improves both MRR and accuracy values. There is an appreciable improvement in the metric values when only Term match (TMatch) feature is used.

We evaluated our individual classifier models and the hierarchical classifier model on Precision, Recall and F1 score metrics.

Table 2 presents the results from the Suggestion classifier. We can clearly see that there is a significant improvement in recall and f1-score when clue words (Clue) are used along with modal verbs (MV) and imperative mood expressions (IME). When clue words (Clue) and modal verbs (MV) features are used along with typed dependency (TypDep) and informativeness score (InfScore) features the precision score increases at the expense of recall. Finally using all the features together shows significant increase in the recall and f1 score values, indicating that all these features together help in identifying the suggestive characteristics of the comments.

Our model was able to identify the question with a precision of 0.86 and recall of 0.73 with F1-score being 0.79.

Table 3 presents the results of the agree and disagree classifier. We can see that using the N-grams provides good performance trade-off.

Table 4 presents the results of the hierarchical classifier. It is evident from the metric values that with the hierarchical classification the precision, recall and f1 score improves since once classified by one classifier as positive, the comment is not passed to the next classifier.

As one can see, the results of classifier are satisfactory to provide useful insights, even though there is room for improving these classifiers.

## 5.1 Visualization

We built a mobile app to present these insights about the comments and the blog to the author and the readers.

Figure 2a shows the landing page of the app where the author/reader can see the list of the blogs he/she has written or can read. The doughnut chart on the right side of each blog shows the category-wise distribution of the comments made on that blog and the number inside the chart is the total number of comments made on that blog. The legends in the top bar shows the class represented by each colour in the chart.

Figure 2b presents the view when a particular blog is chosen. The blog is partitioned into the segments demarcated by the blocks. The number in parenthesis besides each category is the count of comments on the blog that belongs to the category. The scroll bar is segmented according to the segments in the blog and the colour represents the dominant comment category for that segment.

Figure 2c shows the view of blog when the author/reader click on the scroll bar which takes him/her to the corresponding segment where the list of the comments and the category they belong to can be seen.

The author(or reader) can also look at the comments on the blog belonging to only a particular category by clicking on that category from the top legend bar. It can be seen from Figure 2d that all the segments coloured yellow have comments from question category.

As one can see, our tool can help the authors and readers by providing insights about what type of reactions/comments a given blog is attracting, and which parts of the blogs are responsible for those reactions. One can consider enabling several features using such information, for example, sorting the blogs based on most or least number of comments of a particular type (agreement/disagreement/question/suggestion). Also, one may directly find the segments of blogs which attract specific type of comments (and need not open the blogs individually to see them). One can also create summaries of comments of a given type for a particular segment of blog.

**Table 1.** Results of the Comment Association Model

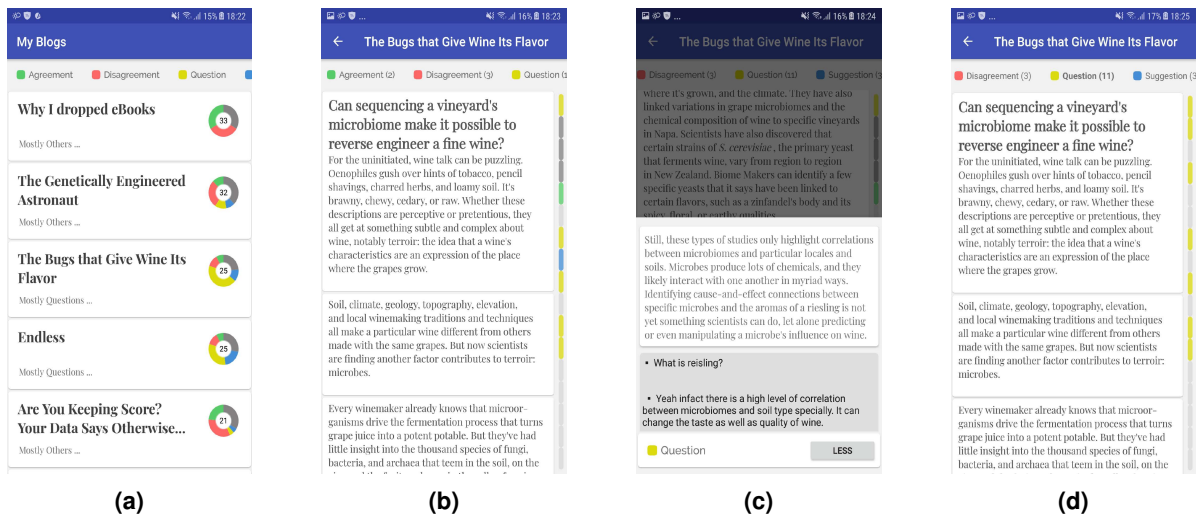
Features	MRR	Accuracy
SegLen+SegPos+EMatch+TMatch+SMatch+LM+W2V	0.745	0.631
SegLen+SegPos+EMatch+TMatch+SMatch+LM	0.763	0.675
SegLen+SegPos+EMatch+TMatch+SMatch+LM+USE	0.769	0.692
TMatch	0.849	0.798

**Table 2.** Results for Suggestion Classifier

Features	Precision	Recall	F1 score
Clue	0.44	0.18	0.25
Clue+MV+IME	0.46	0.64	0.54
Clue+MV+TypDep+InfScore	0.48	0.59	0.53
Clue+MV+IME+TypDep+InfScore	0.47	0.62	0.53

**Table 3.** Results for Agree-Disagree Classifier

Features	Precision		Recall		F1 score	
	Agree	Disagree	Agree	Disagree	Agree	Disagree
LIWC	0.54	0.55	0.43	0.65	0.48	0.60
Glove+N-grams+PosNeg	0.63	0.65	0.63	0.65	0.63	0.65
Glove+N-grams+Pos+Neg	0.63	0.66	0.64	0.65	0.64	0.66
Glove+N-grams+Affin	0.67	0.66	0.60	0.72	0.63	0.69
N-grams	0.65	0.79	0.84	0.58	0.73	0.67

**Fig. 2.** Different Visualization of the Mobile App

**Table 4.** Results for Hierarchical Classifier

Class	Precision	Recall	F1 score
Suggestion	0.47	0.62	0.53
Question	0.95	0.75	0.84
Agreement	0.63	0.85	0.72
Disagreement	0.76	0.50	0.60

Such summaries are more useful as they would be talking about the same thing in same manner, and hence the possibility of creating a coherent summary is higher.

## 6 Conclusions

This paper presents a way to leverage the information present in the comments and deliver insights to the authors about the audience's reaction to the content at a granular level. These insights will help the author in understanding the audience and improving upon the future content. Readers can also benefit from the comments as they help in understanding the blog and possibly help in prioritizing which blog (and which parts of it) they should read.

## References

1. Abbott, R., Walker, M., Anand, P., Fox Tree, J. E., Bowman, R., & King, J. (2011). How can you say such things?!?: Recognizing disagreement in informal political argument. *Proceedings of the Workshop on Languages in Social Media*, Association for Computational Linguistics, pp. 2–11.
2. Chu, T., Jue, K., & Wang, M. (2016). Comment abuse classification with deep learning. Von <https://web.stanford.edu/class/cs224n/reports/2762092.pdf> abgerufen.
3. Dang, Y., Zhang, Y., & Chen, H. (2010). A lexicon-enhanced method for sentiment classification: An experiment on online product reviews. *IEEE Intelligent Systems*, Vol. 25, No. 4, pp. 46–53.
4. De Marneffe, M.-C., MacCartney, B., Manning, C. D., et al. (2006). Generating typed dependency parses from phrase structure parses. *Proceedings of LREC*, volume 6, Genoa Italy, pp. 449–454.
5. Go, A., Bhayani, R., & Huang, L. (2009). Twitter sentiment classification using distant supervision. *CS224N Project Report, Stanford*, Vol. 1, No. 12.
6. Hou, L., Li, J., Li, X., Qu, J., Guo, X., Hui, O., & Tang, J. (2013). What users care about: A framework for social content alignment. *IJCAI*, pp. 1401–1407.
7. Hou, L., Li, J., Li, X.-L., Tang, J., & Guo, X. (2017). Learning to align comments to news topics. *ACM Transactions on Information Systems (TOIS)*, Vol. 36, No. 1, pp. 9.
8. Hu, M. & Liu, B. (2004). Mining and summarizing customer reviews. *Proceedings of the tenth ACM SIGKDD international conference on Knowledge discovery and data mining*, ACM, pp. 168–177.
9. Lai, M., Fariás, D. I. H., Patti, V., & Rosso, P. (2016). Friends and enemies of clinton and trump: using context for detecting stance in political tweets. *Mexican International Conference on Artificial Intelligence*, Springer, pp. 155–168.
10. Manning, C. D., Surdeanu, M., Bauer, J., Finkel, J., Bethard, S. J., & McClosky, D. (2014). The Stanford CoreNLP natural language processing toolkit. *Association for Computational Linguistics (ACL) System Demonstrations*, pp. 55–60.
11. Misra, A. & Walker, M. (2017). Topic independent identification of agreement and disagreement in social media dialogue. *arXiv preprint arXiv:1709.00661*.
12. Negi, S., de Rijke, M., & Buitelaar, P. (2018). Open domain suggestion mining: Problem definition and datasets. *arXiv preprint arXiv:1806.02179*.
13. Nielsen, F. Å. (2011). A new anew: Evaluation of a word list for sentiment analysis in microblogs. *arXiv preprint arXiv:1103.2903*.
14. Pennebaker, J. W., Francis, M. E., & Booth, R. J. (2001). Linguistic inquiry and word count: Liwc 2001. *Mahway: Lawrence Erlbaum Associates*, Vol. 71, No. 2001, pp. 2001.
15. Pennington, J., Socher, R., & Manning, C. (2014). Glove: Global vectors for word representation. *Proceedings of the 2014 conference on empirical methods in natural language processing (EMNLP)*, pp. 1532–1543.
16. Savigny, J. & Purwarianti, A. (2017). Emotion classification on youtube comments using word embedding. *Advanced Informatics, Concepts, Theory, and Applications (ICAICTA), 2017 International Conference on*, IEEE, pp. 1–5.

17. Severyn, A., Moschitti, A., Uryupina, O., Plank, B., & Filippova, K. (2014). Opinion mining on youtube. *Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, volume 1, pp. 1252–1261.
  18. Sil, D. K., Sengamedu, S. H., & Bhattacharyya, C. (2011). Readalong: reading articles and comments together. *Proceedings of the 20th international conference companion on World wide web*, ACM, pp. 125–126.
  19. Sil, D. K., Sengamedu, S. H., & Bhattacharyya, C. (2011). Supervised matching of comments with news article segments. *Proceedings of the 20th ACM international conference on Information and knowledge management*, ACM, pp. 2125–2128.
  20. Walker, M. A., Tree, J. E. F., Anand, P., Abbott, R., & King, J. (2012). A corpus for research on deliberation and debate. *LREC*, pp. 812–817.
  21. Wicaksono, A. F. & Myaeng, S.-H. (2012). Mining advices from weblogs. *Proceedings of the 21st ACM international conference on Information and knowledge management*, ACM, pp. 2347–2350.
  22. Wicaksono, A. F. & Myaeng, S.-H. (2013). Automatic extraction of advice-revealing sentences for advice mining from online forums. *Proceedings of the seventh international conference on Knowledge capture*, ACM, pp. 97–104.
- Article received on 06/02/2019; accepted on 04/03/2019.  
Corresponding author is Abhilasha Sancheti.

# Word Embeddings for IoT Based on Device Activity Footprints

Kushal Singla, Joy Bose, Nitish Varshney

Samsung R and D Institute, Bangalore,  
India

{kushal.s, nitish.varshney}@samsung.com, joy.bose@ieee.org

**Abstract.** With the expansion of IoT ecosystem, there is an explosion of the number of devices and sensors and the data generated by these devices. However, the tools available to analyze such data are limited. Word embeddings, widely used in the natural language processing (NLP) domain, provides a way to get similar words to the current word. In this paper, we extend the theory of word embeddings to the area of IoT devices, proposing a method to generate the word embeddings for IoT devices and sensors in a smart home based on their activity. We model IoT devices as vectors using a concept like Word2Vec and App2Vec, where the time between the device firings is also taken into account. These computed word embeddings can be used for a variety of use cases, such as to find similar devices in an IoT device store, or as a signature of each type of IoT device. We show results of a feasibility study on the CASAS dataset and a private real-world dataset of IoT device activity logs, using our method to identify the patterns in embeddings of various types of IoT devices in a household. We get a probability of more than 0.65 for similar types of devices clustering together, independent of session gap value and embedding vector size for the CASAS dataset. We also get a probability of 0.4 on the private dataset, independent of session gap value and embedding vector size.

**Keywords.** Word2Vec, IoT2Vec, word embeddings, smart home, internet of things, natural language processing.

## 1 Introduction

The Internet of Things (IoT) has grown exponentially in recent times, with IoT sensors and devices being used in many real-life use cases such as smart homes. These sensors generate lots of data each second.

Analysis of the data generated by the IoT sensors and devices in a smart home can lead to valuable insights about the usage habits and the devices themselves. However, the number of studies on real-life smart home IoT datasets to get insights about the device usage patterns is limited.

The popular Word2Vec model [1] provides ways to generate word embeddings based on the usage of the words in one or more documents. Here, an IoT usage log can be considered as a document, and the logs of a given IoT device within a given small time window can be treated as a word. Such word embeddings can be used, for example, to find similarity between two documents. App2Vec [2] is a modified adaptation of Word2Vec for apps based on app behavior, with additional weightage based on time of firings.

Using an approach like Word2Vec and App2Vec, we attempt to create embeddings for IoT devices based on their usage, using data obtained from the usage logs of the devices. These embeddings can be used, for example, to find similar IoT devices for a given device. Such a model we call IoT2Vec.

In this paper, we describe the generation of embeddings using some publicly and privately available IoT datasets and describe some principles how this can be adapted for different IoT device usage data. We create a model to take the device usage data as input, create embeddings for the devices and identify a new device of the same type based on similar usage data. There can be several applications for such a model to create embeddings and find similar IoT devices.

One application of such a model is to make a search function to search for similar devices in the

vicinity. Using this method, defective devices can be replaced based on their function.

If we know the footprint of the IoT device, we can identify which another device is best suited to replace it. This can also be used to recommend similar IoT devices from different vendors, such as in an online e-store. Another useful application for such a model is routine disruption due to faulty/malfunctioning devices, where a faulty/malfunctioning device in the routine can be replaced by similar devices in the vicinity. It can also be used to transfer a given user's routine from one location to another in case the user has changed their home location or gone for an outing or leisure travel.

An-other common application can be to build a location classifier based on IoT de-vices in that location. For example, given that a pub usually has dim lights, it is likely that another pub will also have similar light settings. Therefore, knowing the IoT devices and their footprint in each location, we can identify the type of location. Another useful application is routines identification, which is the primary enabler for automating user's action in a smart home.

Automatically identified routines allow users to control and automate many aspects of a home without user intervention and without going into unnecessary hassle of creating home automation recipe for themselves. Especially for elder persons living alone at home, it is very difficult as they need to be aware of their own routine, variations followed to pursue routine and technical knowledge of the relevant home automation devices in home.

The rest of this paper is organized as follows: in the following section we survey approaches to current approaches related to identifying similar IoT devices. Section 3 describes the theory and method which we use to generate the embeddings. Section 4 gives the results of our method applied on public and private IoT datasets, along with validation and real-life use cases. Section 5 concludes the paper.

## 2 Related Work

There are a few instances of related work in applying machine learning to find similar IoT devices.

Xu [3] proposed a system for searching for and finding similar IoT devices as a result of user queries, based on a similarity measures based on the semantic and other properties of the IoT objects such as the object location. Kang [4] suggested various methods to identify correlations between IoT devices, including attributes such as location, usage count, sensor list, service name etc. They also suggested using Word2Vec model to calculate the adjacency between IoT devices. However, they did not provide concrete details on exactly how the vectors would be calculated and the issues involved when working with real datasets.

Tian [5] mentioned a mechanism to automatically collect security related information from an IoT app. Palit [6] mentioned a system to identify IoT resource requirements such as sensor accesses from service descriptions, using NLP techniques to parse the Android app descriptions to determine which sensors were required. Hong [7] used similarity measures between IoT devices to provide context aware services to users. Truong [8] proposed a method for searching similar IoT sensors, computing a similarity score based on fuzzy sets.

The patent of Derek Lin [21] describes analysis of device similarity using methods such as principal components analysis. However, most of these approaches require prior information about the IoT devices, such as the parameters needed to determine similarity. Such prior information might not be readily available.

Word embeddings have been used for a variety of use cases including product and item recommendations [15, 16], similar nodes in a network [17], and multi-media [18-20]. However, they have not been applied to determine similarity for IoT devices as of late. In this paper, we use IoT sensor or device usage patterns to create the word embeddings and identify the type of IoT device. Having such an approach has the advantage of not needing prior information about the devices, only their usage data is needed instead.

## 3 Model for Generation of Word Embeddings

Most IoT devices are used in certain patterns that repeat over time. Similar kinds of IoT devices will

have similar activity footprints. The IoT device type can be identified by the pattern of usage.

The time of usage and location of devices also carries useful information. A model can be trained to encode this pattern as word embeddings, which will help to identify the IoT device that has similar patterns.

### 3.1 Theory of Word Embeddings for IoT Devices

In the Word2Vec model [1], a neural network is trained to map a word to a vector in such a way that the probability of predicting a word (target) given a context of words (in CBOW model) is maximized. The model is trained on a large dataset of documents, so it is expected to capture all possible variations and patterns in which the words are used together. So it aims to maximize the following log likelihood for all words in the dictionary:

$$\log P(W_i|\text{context}) = \frac{\text{similarity}(W_i|\text{context})}{\sum_k \text{similarity}(W_k|\text{context})}, \quad (1)$$

where  $W_i$  is the word vector corresponding to the  $i$ th word, and similarity between vectors is measured by cosine distance. For example if the model is trained on the sentence 'the cat is on the table', the aim is to maximize the probability of predicting 'cat' when the following words are presented to the model 'the \_\_\_\_ is on the table'. In the Skip-Gram model, on the other hand, the objective is to present a word and predict its surrounding words in a given window (or context length). So if the word 'cat' is presented as input, the model should learn to predict 'the, is, on, the, table' as output.

In this paper, for IoT devices and sensors, we aim to create a word vector for each unique IoT device such that its activity can be predicted given its context. If sensor 1 fires along with sensor 2 and 3, then given the sensor 2 and 3 firing, the model should be able to predict sensor 1. In this sense, it is like the Word2Vec model. We train our model on a dataset of IoT device activities and hope to capture the patterns of cooccurring sensor activity for different types of sensors.

In our approach, we define an IoT device session sequence as being similar to the app sequence defined in the App2Vec paper: If  $D_1, D_2,$

$D_3$  are three IoT devices or sensors in a household, and  $g_1, g_2, g_3$  etc. are the time gaps between the transition times of these devices, then an example usage session can be represented as  $(D_1, g_1, D_2, g_2, D_3, g_3, D_1, g_4, D_2)$ .

Here, we define a session as a certain length of time, say 60 seconds or 600 seconds. We only consider the transitions of the sensors (OFF to ON and ON to OFF for binary states, or a defined range of values for bins in case of sensors like thermostat) for our purpose. Within a session, the aggregate activities of all the sensors in sequential order  $(D_1, D_2, D_3, D_1, D_2)$  is a sentence and the activity of any single device or sensor ( $D_1$ ) is a word. After getting the words and sentences for all the sessions in our dataset, we analyze them to create the embeddings vectors using the Word2Vec or App2Vec method.

We have two choices regarding the time gap between sensor activations ( $g_1, g_2, g_3$ ):

1. Ignore the time gap and consider only the order of transitions of devices within a session  $(D_1, D_2, D_3, D_1, D_2)$  when creating the sentence and words for the embeddings vector.
2. Consider the time gap and introduce a weight  $x^t$  for the context words in the CBOW model, where  $t$  is the time gap in minutes from the target word and  $x < 1.0$  is a similarity factor that decays with time difference between the activation of the target device and context device. The App2Vec model [2] used a similar weightage concept (with  $x$  empirically determined as 0.8) to determine the similarity of app usage vectors. The idea is that if two IoT devices have a small time gap, their vectors should be more similar than if two devices have a larger time gap.

### 3.2 Method to Create a Word Embeddings Vector for IoT

Using the previously discussed theory, we define some steps to generate and analyze the word embeddings from IoT device sensor logs.

Our method includes the following steps:

1. Filter out the IoT sensors whose data is not meaningful or we cannot make sense of the data.

2. Examine the activity data of the selected sensors to see whether it shows meaningful activity or actions.
3. Extract only the values where the sensor state is in transition (e.g. ON to OFF or OFF to ON).
4. Build a session of the sensor values (similar to sentence in NLP domain) by choosing a session gap. Session gap is the gap of time where we construct the boundaries of each session. Within a session, we consider both the discussed approaches (a) only consider the order of firings of different devices or sensors. The exact time gap between firings within a session is ignored and (b) set a weightage  $x^t$  where  $x < 1.0$  and  $t$  is the time gap in minutes.
5. Once the sessions are defined, we treat each session as a sentence and the device Id as a word. Each sentence will contain a sequence of IoT device Ids such as (M008, M009, D010) which is the order of firings of the devices within the session.
6. Train the lot2Vec model using the session data extracted from the dataset. The input to the training model is the document comprising of the created sentences in the previous step. The output of the model is the embedding vector for each type of sensor or device. We can select a certain dimension, such as 100, for the size of the vector embeddings.
7. Compute the similarity between the vector embeddings of each sensor/device with the other sensors. Furthermore, we perform dimensionality reduction and construct a t-SNE plot for easier visualization of the sensor activities in terms of contextual similarity, i.e. which IoT devices or sensors are being activated together.
8. Visually examine the t-SNE plots to detect patterns of similarity in the activity data for each type of IoT device or sensor with other sensors. Following the above steps, the embeddings vector of a given device or sensor type can be generated from its activity logs.

Table 1 shows the algorithm to generate the word embeddings with and without the weighed similarity factor.

**Table 1.** Algorithm to identify the embeddings vector for a given device

**Input:** Device activation sequence for the devices  $D_1, D_2, \dots, D_n$

**Output:** Embeddings vector for devices  $D_1, D_2, \dots, D_n$

1. Break the device activation sequence into sessions for a given value of session gap, considering only device state transitions. The session represents a sentence.
2. Train a model using CBOW, similar to word2vec, using the generated session sequences
3. Once training is completed, the embeddings for each device are generated
4. Repeat the steps 2-3, using a weighed CBOW model with weight  $x^t$  where  $t$  is the time gap in minutes between device activations and  $x$  is a similarity factor  $< 1$ .
5. **exit:** end procedure

**Table 2.** Algorithm to identify device type from activity logs

**Input:** Stored device embeddings for different device or sensor types  $D_1 (E_1), D_2 (E_2)$  etc.

**Output:** Device type of a new device  $D_i$  given its usage data

1. Generate embeddings vector  $E_j$  from the usage data of the new device  $D_j$
2. Compute the similarity of the embedding vector  $E_j$  with each of the stored embedding vectors  $E_1, E_2 \dots$
3. Find the device  $D_i$  whose similarity value of the embedding vector  $E_i$  with  $E_j$  is highest and above a threshold
4. Define the device type of  $D_j$  as equal to the device type of  $D_i$
5. **exit:** end procedure

### 3.3 Method to Identify the Device Type of a New or Unknown IoT Devices from its Usage Logs

The table 2 shows the algorithm for identifying the device type of a new or unknown IoT sensor or



device from its activity logs, once we have stored device embeddings of a set of devices. The principle is to generate the embeddings vector for the new device and determine which of the stored embeddings is closest to the generated embeddings vector.

## 4 Experimental Details and Results

For our experiments to validate our method and to explore the possibility of generating embeddings for IoT devices, we needed one or more datasets that would provide us with data from multiple IoT devices in the same locations over a period of time.

For our purpose, we tried several candidate datasets [9-12] and finally chose a dataset 20 from the Kyoto dataset list of CASAS [11, 12]. This dataset has 2 years' worth of data from a household consisting of two residents, with various IoT devices including motion sensors, doors (fridge, freezer, and microwave), shelves etc. Each data item consists of the following fields: time, sensor name, sensor state.

Fig. 1 shows an extract from a layout of a room in a house in the CASAS Kyoto dataset 20 [11, 12].

We then created word embeddings for various devices in the dataset for which we have data. We then used this embeddings to identify the devices. We analyzed the dataset in Spark and used Word2Vec to find patterns in the data.

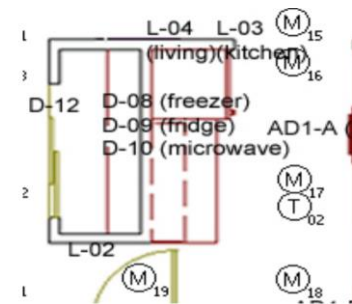
During the preprocessing step, we ignored the light sensor, gyro sensor and a few others, since they were firing without any discernible patterns. We selected the following sensors for analysis: Motion sensor, door sensor, item sensor, shake sensor, fan sensor, experimental switch.

We then obtained a sequence of sensor states, belonging to multiple sensors, ordered by time. We ignored the actual time of sensor state change and only noted the sequence.

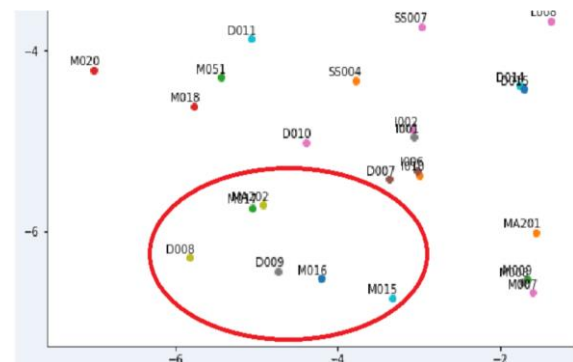
Our objective, as mentioned earlier, was to determine the similarity between different sensors on the basis of their activity.

In our chosen dataset [11, 12], the device D008 is a door sensor corresponding to a freezer door, where the freezer is located in the kitchen.

The similarity between vector embeddings that we obtained for this D008 sensor for the 60 second session gap is as below:



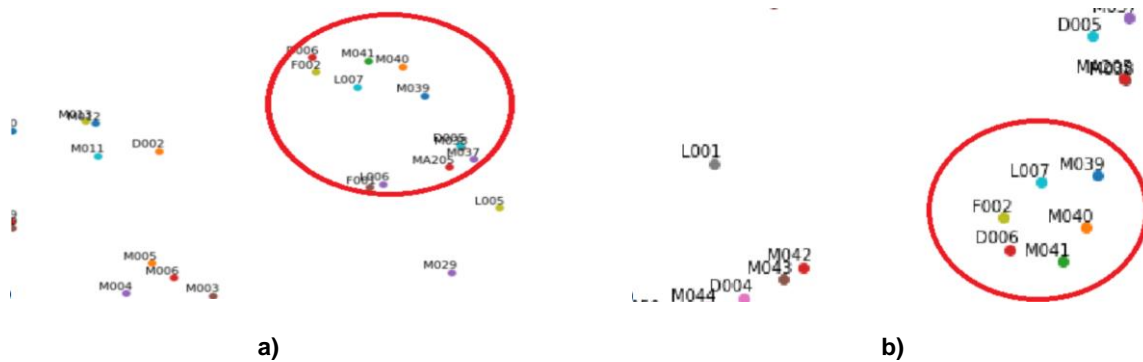
**Fig. 1.** Extract from the layout of a room in the CASAS Kyoto dataset [11], showing how some motion sensors (beginning with M) and doors (beginning with D) are located close together in the kitchen



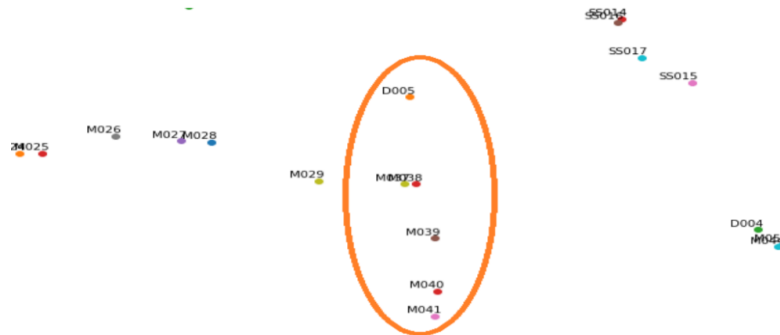
**Fig. 2.** Extract from the t-SNE plot of the activity of sensors from the CASAS Kyoto dataset [10], using 60 seconds as the interval, showing the contextual proximity of the motion sensors close to the kitchen (M015, M016) and door sensors of fridge (D009) and freezer (D008)

D008 [('M017', 0.49945521354675293),  
('M016', 0.48164984583854675),  
('MA202', 0.4487079977989197),  
('M018', 0.4332207143306732),  
('D009', 0.41653889417648315),  
('D015', 0.3721662163734436),  
('M015', 0.3238069415092468),  
('M051', 0.2985246777534485),  
('D010', 0.2684941589832306),  
('D014', 0.24952027201652527)].

From the above similarity between vector embeddings, we can derive the same conclusion, that door sensor D008 is close to motion sensors M017 and M016 which are in close proximity.



**Fig. 3.** (a) Extract from the t-SNE plot for Kyoto-20 Dataset (b) Sensor activity for 600 seconds gap. Here, sensors located near the toilet M038, M039, M040, M041, D006, D005 show similar patterns of activity across session gaps



**Fig. 4.** Extract from the t-SNE plot of the activity of sensors from the CASAS Kyoto-20 dataset [11], using 600 seconds as the interval and time decay factor weight  $x^t$  for  $x=0.9$  and  $t$  being the time gap in multiples of 15 sec, showing the contextual proximity of the motion sensors close to the toilet (M038 to M041) and door sensor D005



Examining the figure 2, we see that door sensor D008 (freezer, located in the kitchen) comes contextually close to motion sensors M015 and M016, located close to the kitchen, and the door sensor D009 of the fridge, also located in the kitchen. Looking at the layout of the house in fig. 1, we see that the sensors D008 and D009 are located in the kitchen and motion sensors M015, M016, M017 are located close to the kitchen. We can explain their contextual similarity as follows: when a person comes into the kitchen, they would activate the motion sensors close to the kitchen, after which they would open the fridge and freezer to get or make some food.

Based on this, we conclude that it is feasible to identify IoT devices based on their contextual similarity. In the following subsections, we analyze a few trends of device activity based on various parameters.

#### 4.1 IoT Device Activity for a Given Value of Session Gap, with Time Gap Ignored

In this subsection, we attempt to find some trends in IoT device activity plotted using the t-SNE method using our word embeddings computed for various devices.

First, we determine whether the IoT device activity across different session gaps shows any significance. For example, it is possible that when we choose a small gap such as 10 seconds, device A and B are close, however when we increase the time gap to say 600 seconds, device A and C are close. So, we visually inspect the T-SNE plots [13] of the activity data to see if that can indeed be the case, or whether similar devices always cluster together.

Fig. 3(a) shows the device activity and sensor locations near the toilet area for a 10 seconds session gap. On visual inspection, we can see that motion sensors M039, M040, M041, D006, D005 etc. show correlated activity patterns. This is also confirmed from a look at the similarity measure of distance, where the vector embeddings for these sensors show closest Euclidean similarity between each other.

Fig. 3(b) shows the same sensors activity for a 600 seconds gap for a session. We see that the same sensors that were active together for a 10

second session gap are also active for a 600 seconds session gap.

Hence, we conclude that for this choice of sensors, the proximity of location (all these sensors are in the toilet area) translates into contextual proximity as well, regardless of session intervals. This could be because whenever someone uses the toilet, the motion sensors and door would always be triggered together. However, for a different choice of sensors, this might not be the case and the timer could be a factor in deciding which sensors trigger together.

#### 4.2 IoT Device Activity, Weighing for Time Gaps within a Session

In this subsection, we repeat the previous experiment for session gap 60 seconds but weighing for time gaps within a session. We choose a session gap of 600 seconds, and time decay factor weight  $x_t$  for  $x = 0.9$  and  $t$  measured in multiples of 15 seconds. The results are shown in fig. 4.

We can see that here too we get the same trend as when ignoring time gaps: the sensors in close proximity M38 to M41 also show closeness in the t-SNE plot.

Perhaps repeating the experiment in the future for a larger time gap (to allow for the distance to be more pronounced when the sensors fire further apart) and varying the  $x$  and  $t$  parameters might show more interesting observations in trends.

#### 4.3 IoT Device Activity for Varying Datasets for the Same Sensor Type, with Time Gap Ignored

In this analysis, we seek to learn if the co activity of different types of sensors is sustained across different datasets.

For this experiment, we chose the Kyoto 11 dataset which was part of CASAS [11, 12]. It had the same layout as the Kyoto 20 dataset that we used earlier. However, the year the data was collected is different. The sensors M40, M41, door sensor D006 are located in the toilet-cum- bathroom.

As earlier, we plotted the word embeddings for the sensors with 60 seconds and 600 seconds gap. The plots are shown in Fig. 5. As we can see, here

too the sensors near the toilet area have similar activity as before. Although a larger study is needed, we can postulate from this data that it is feasible to believe that the patterns of sensor co-activity as per contextual similarity can hold across datasets.

#### 4.4 Validation: Probability of Similar Device Types Clustering Together

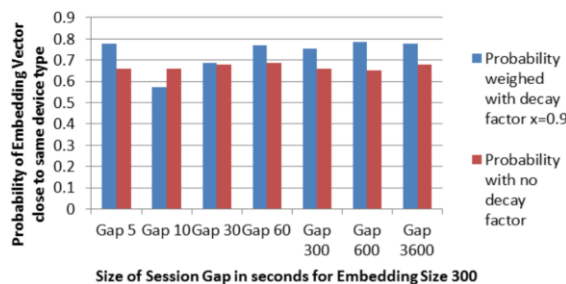
In order to validate the usefulness of our IoT2Vec model to predict similar devices, we also calculated and plotted the vector similarity of the embedding vectors of similar devices from other devices of the same device type. For each unique device, we first determine the closest device, then determine the type of the closest device. We compute the ratio of positive matches Vs the number of total devices and plot this across different session gap values. We repeat this experiment with and without the weighed decay factor, as explained in earlier sections.

The result is plotted in Fig. 6 for an embedding size of 300. As we can see, the probability is higher than 0.65 across session gaps for similar types of devices clustering together, thus validating our approach. The session gap of 600 seconds produced the best results. The probabilities with a weighed decay factor of 0.9 (where longer time differences between device activations is penalized with a decay factor) were higher across session gaps than the probabilities with no decay factor, indicating that most device activations appear close together in time.

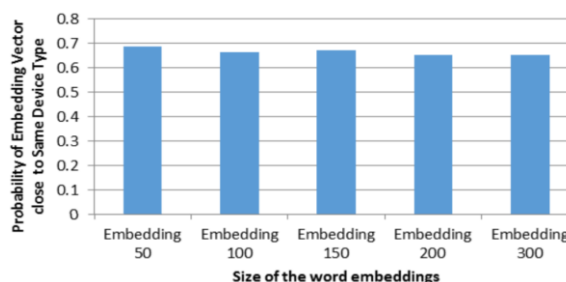
Fig. 7 is a plot of the probability when the word embedding size is varied from 50 to 300, for a session gap of 600s. As we can see, the choice of word embedding size does not affect the probability significantly. Regardless of the embedding size, we get more than 0.65 probability that similar types of devices have similar word embedding vectors.

#### 4.5 Private Dataset Validation: Validating the IoT2Vec Model on Real Life Use Cases

As part of developing a home automation solution, our organization has collected smart home users' data.



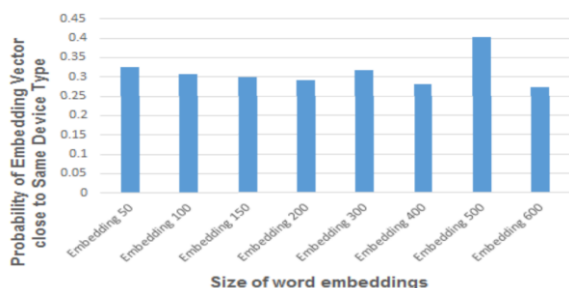
**Fig. 6.** Graph showing the probability of the word embeddings vectors coming close to the same device types, for an embedding size of 300, plotted with and without the decay factor



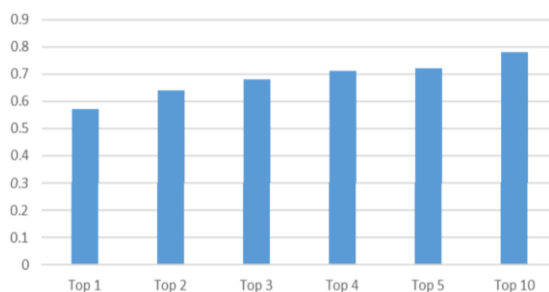
**Fig. 7.** Graph showing the probability of the word embeddings vectors coming close to the same device types, for a gap of 600 seconds, plotted with and without the decay factor

The dataset has 2 weeks' worth of data, which has been collected from smart home flats of 17 different users.

These smart homes had a mixture of single users and couples, with a variety of IoT sensors and appliances including Monoprice Z-Wave Plus Door/Window Sensor, Nest Weather, Switch, Smoke Detector, Smart Plug, TED5000, uDTH, Arlo Pro Basestation Siren, Evohome Heating Zone, Camera, Moisture Sensor, Rachio Zone, Aeon Key Fob, Smart Lock, Samsung SmartCam, Network Audio, Light, Hue Dimmer Button Controller AB, Lightwave On Off Device, Weather, Simulated Contact Sensor, Aeon Minimote, Motion Sensor, Z-Wave Device Multichannel, Remotec ZRC-90 Scene Master, Z-Wave Switch Generic, LAN Hue White Ambiance Bulb, Lightwave Dimmer Switch, LIFX Color Bulb, Door Bell, Multi-functional Sensor and Logitech Harmony Hub



**Fig. 8.** Graph showing the probability of the word embeddings vectors coming close to the same device types, for an embedding size of 50 to 600, plotted with the decay factor, for the private dataset



**Fig. 9.** Graph showing the probability of minimizing user discomfort in case of Routine Disruption, for the private dataset. The X axis plots K, where the top K similar devices are identified, and Y axis plots the probability of minimizing user discomfort

C2C. In this dataset, the device data is time ordered and is represented by a sensor event tuple as follows: <timestamp;masked device id;device state; device category>

This dataset is more significant than the previous (public) dataset, since the variety of the users and the complexity of the devices / appliances and number of possible device states is higher. In addition, the home layout is unknown, so we can only find information on the type of device along with state transitions, and not where it is physically situated.

However, our organization has extracted routines for these users, which are available to us. Fig. 8 is a plot of the probability when the word embedding size is varied from 50 to 600, for a session gap of 600s for the private dataset. We can see that the embedding size of 500 provides the best results. We have utilized this dataset to

validate the usefulness of the IoT2Vec model on real life use cases. One of these real-life use cases where we applied the solution was identifying alternative for faulty/malfunctioning devices, which are part of the user's routine.

Whenever any device, which is part of the user's daily routine, becomes faulty or malfunctioning, it would lead to the user's discomfort. Therefore, there is a need to find an alternative device, which can be used to act as a replacement of malfunctioning device. To evaluate the usefulness of our IoT2Vec model in this case, we tried to identify the replacements for devices (randomly assuming one of the devices are faulty/malfunctioning) in user's routine. For each user routine, we first randomly choose a device and assume it is faulty. Then we try to identify contextually similar top-K devices to the faulty device.

The identified contextually similar device is plugged into the unmodified user routine and provided to the routine identification team for evaluation. The assumption here is that using the modified routine (with the replacement device), the user should be able to perform his or her daily tasks, which the same user was originally performing using unmodified routines, without any discomfort. Fig. 9 is a plot of the probability of minimizing user discomfort in case of routine disruption by providing a new user routine after replacing faulty/malfunctioning device with top-k alternatives.

As we can see, the probability of minimizing user discomfort increases with the increase in the number of closest devices matched for similar type. However, after closest devices is set to 3, it does not increase any further. This is because for some devices such as a refrigerator, the user does not have alternate devices in the house which can be used as replacements in case of some fault in the original device.

## 5 Conclusion

We have proposed a method to generate word embeddings for IoT devices, based on their usage patterns. We showed that IoT devices in similar areas in a given household can be found to have similar usage patterns. We get a probability of at

least 0.65 similar types of devices clustering together regardless of the session gap or word embedding size chosen. We found that the session gap does not affect the similarity, whereas using a decay factor showed a higher value of clustering similarity. Thus, it is feasible to recognize IoT devices based on the embeddings.

In future, we plan to focus on smart city scenarios. We plan to build a location classifier based on IoT devices used in that location and embedding similarity could capture the location type. We also plan to focus on activity generated by smart fridge and TV, to get higher level understandings of the patterns.

We also plan to generate routines using the IoT2Vec model. In such a scenario, rather than creating a word vector for each unique IoT device, we will generate sentence vectors using Sentence2Vec [22] for each session. Further, clustering could be utilized to identify recurring patterns in the data. Then, routines can be generated from each segment cluster using CLIQUE [23]. Another such routine generation approach we plan to evaluate is by Chanda et. al [24] who used NLP techniques such as language modeling to recommend routines.

## References

1. Mikolov, T., Chen, K., Corrado, G., & Dean, J. (2013). *Efficient estimation of word representations in vector space*. arXiv preprint arXiv:1301.3781. pp. 1–12.
2. Ma, Q., Muthukrishnan, S., & Simpson, W. (2016). App2vec: Vector modeling of mobile apps and applications. *IEEE/ACM International Conference on Advances in Social Networks Analysis and Mining (ASONAM)*, pp. 599–606. DOI: 10.1109/ASONAM.2016.7752297.
3. Xu, W. (2015). Modeling and exploiting the knowledge base of web of things. *Ubiquitous Computing*. Université Pierre et Marie Curie - English. ffNNT : 2015PA066009ff.
4. Kang, H., Kim, M., Kwon, S., & Kim, N.S. (2016). A Design of IoT Device similarity vector based workflow management system. *International Conference on Information and Communication Technology Convergence (ICTC)*, pp. 1036–1038. DOI: 10.1109/ICTC.2016.7763361.
5. Tian, Y., Zhang, N., Lin, Y.H., Wang, X., Ur, B., Guo, X., & Tague, P. (2017). Smartauth: User-centered authorization for the internet of things. *26th USENIX Security Symposium*, pp. 361–378.
6. Palit, A., Srivatsa, M., Ganti, R., & Simpkin, C. (2017). Identifying sensor accesses from service descriptions. *IEEE International Conference on Big Data (Big Data)*, pp. 3006–3011. DOI: 10.1109/BigData.2017.8258271.
7. Hong, I. & Lee, Y. (2016). Key-Device based place recognition using similarity measure between IoT spaces. *IEEE International Conference on Smart Computing (SMARTCOMP)*, pp. 1–5. DOI: 10.1109/SMARTCOMP.2016.7501701.
8. Truong, C., Römer, K., & Chen, K. (2012). Sensor similarity search in the web of things. *IEEE International Symposium on a World of Wireless, Mobile and Multimedia Networks (WoWMoM)*, pp. 1–6. DOI: 10.1109/WoWMoM.2012.6263791.
9. NOAA (2018). <http://tidesandcurrents.noaa.gov/gmap3>
10. Intel lab (2018). <http://db.csail.mit.edu/labdata/labdata.html>
11. WSU Casas (2018). <http://ailab.wsu.edu/casas/datasets/>
12. Cook, D.J., Crandall, A.S., Thomas, B.L., & Krishnan, N.C. (2013). CASAS: A Smart Home in a Box. *Computer*, Vol. 46, No. 7, pp. 62–69. DOI:10.1109/MC.2012.328.
13. Maaten, L.V.D. & Hinton, G. (2008). Visualizing data using t-SNE. *Journal of machine learning research*, pp. 2579–2605.
14. Singla, K. & Bose, J. (2018). IoT2Vec: Identification of Similar IoT Devices via Activity Footprints. *International Conference on Advances in Computing, Communications and Informatics (ICACCI)*, pp. 198–203. DOI: 10.1109/ICACCI.2018.8554398.
15. Vasile, F., Smirnova, E., & Conneau, A. (2016). Meta-prod2vec: Product embeddings using side-information for recommendation. *10th ACM Conference on Recommender Systems*, pp. 225–232. DOI: 10.1145/2959100.2959160.
16. Ozsoy, M.G. (2016). *From word embeddings to item recommendation*. arXiv preprint arXiv:1601.01356.
17. Grover, A. & Leskovec, J. (2016). node2vec: Scalable feature learning for networks. *22nd ACM SIGKDD International Conference on Knowledge discovery and data mining*, pp. 855–864. DOI: 10.1145/2939672.2939754.
18. Wang, D., Deng, S., Liu, S., & Xu, G. (2016). Improving music recommendation using distributed representation. *Proceedings of the 25th International Conference Companion on World Wide Web*

- (WWW) Conference, pp. 125–126. DOI: 10.1145/2872518.2889399.
19. **Habibian, A., Mensink, T., & Snoek, C.G. (2016).** Video2vec embeddings recognize events when examples are scarce. *IEEE transactions on pattern analysis and machine intelligence*, Vol. 39, No. 10, pp. 2089–2103. DOI: 10.1109/TPAMI.2016.2627563.
  20. **Klein, B., Lev, G., Sadeh, G., & Wolf, L. (2015).** Associating neural word embeddings with deep image representations using fisher vectors. *IEEE Conference on Computer Vision and Pattern Recognition*, pp. 4437–4446.
  21. **Lin, D., Kaufman, A., & Villa, Y. (2016).** *Patent and Trademark Office*.
  22. **Le, Q. & Mikolov, T. (2014).** Distributed representations of sentences and documents. *International conference on machine learning*, pp. 1188–1196.
  23. **Dehghan, A., Modiri-Assari, S., & Shah, M. (2015).** Gmmcp tracker: Globally optimal generalized maximum multi clique problem for multiple object tracking. *IEEE Conference on Computer Vision and Pattern Recognition*, pp. 4091–4099.
  24. **Chanda, P.K., Varshney, N., & Subash, A. (2017).** Applications of Natural Language Techniques in Solving SmartHome Use-Cases. *Natural Language Processing and Information Systems (NLDB)*, pp. 214–217.

Article received on 25/01/2019; accepted on 04/03/2019.  
Corresponding author is Kushal Singla.





# Towards Simple but Efficient Next Utterance Ranking

Basma El Amel Boussaha, Nicolas Hernandez, Christine Jacquin, Emmanuel Morin

Université de Nantes,  
France

firstname.lastname@ls2n.fr

**Abstract.** Retrieval-based dialogue systems converse with humans by ranking candidate responses according to their relevance to the history of the conversation (context). Recent studies either match the context with the response on only sequence level or use complex architectures to match them on the word and sequence levels. We show that both information levels are important and that a simple architecture can capture them effectively. We propose an end-to-end multi-level response retrieval dialogue system. Our model learns to match the context with the best response by computing their semantic similarity on the word and sequence levels. Empirical evaluation on two dialogue datasets shows that our model outperforms several state-of-the-art systems and performs as good as the best system while being conceptually simpler.

**Keywords.** Dialogue systems, response retrieval, sequence similarity.

## 1 Introduction

Recently, many works were interested in building neural dialogue systems that converse with humans in natural language by either generating or retrieving responses. Despite the capacity of generative systems to produce customized responses for each conversation context, they tend to generate short and general responses [14]. Thus, they prefer to generate, for example *"I don't know"* and *"Good !"*, most of the time. This is due essentially to the lack of diversity in their objective function [9]. On the other hand, response retrieval systems are able to provide more accurate and syntactically correct responses [13, 21] by ranking a set of candidate responses based on their coherence with the context. In this work we focus on this category of dialogue systems.

Context	
A	Hi, I can not longer access the graphical <b>login screen</b> on ubuntu 12.04
B	what exactly happen?
A	I can't remember the error message, would it have auto-logged to a file or should I reboot quick?
B	you mean it won't <b>automatically start</b> and what happen then?
A	it just stop at a text <b>screen</b> , but I can access the command line <b>login</b> via alt F1-6, and <b>start x manually</b> there. I think it might me <b>lightdm</b> that's break but I'm not sure
Candidate responses	
R1	for me <b>lightdm</b> often won't start <b>automatically</b> either. It show me console tty1 instead and I have to <b>start lightdm manually</b> ✓
R2	what about sources.list ? ✗

**Fig. 1.** Example of a conversation between two participants (A and B) extracted from the Ubuntu Dialogue Corpus [12]

Given the technical conversation between two users in Figure 1, a response retrieval system should rank the first response before the second one. It is important that the system captures the common information (carried by words written in bold) between the context turns and between the whole context and the candidate response. According to [21], the challenges of the next response ranking task are (1) how to identify important information (words, phrases, and sentences) in the context and how to match this information with those in the response and (2) how to model the relationships between the context utterances.

Most of the recent works use complex architectures to capture sequence and word level

information from the context and the candidate response in addition to multiple response matching and aggregation mechanisms [24, 21]. Other works neglect word level information and simply rank candidate responses based on only sequence level information [12, 6, 2, 20, 22]. Some of them use external modules (ex. topic modelling) or have external knowledge requirements (ex. knowledge bases/graphs), making their training and adaptation to different domains more complex.

In this paper, we argue that these approaches suffer from two fundamental drawbacks: the complexity of their architectures and/or their domain dependency. We propose a simple neural architecture that is domain independent and can be trained end-to-end without any external knowledge. We evaluate our approach on two large dialogue datasets of two different languages: the Ubuntu Dialogue Corpus [12] and the Douban Conversation Corpus [21]. We show that the resulting system achieves state-of-the-art performance while being conceptually simpler and having fewer parameters compared to the previous, substantially more complex, systems.

The remainder of this work is as follows: first, we investigate works around retrieval-based dialogue systems. Second, we describe the problem and the architecture of our system. Third, we present the experimental environment and the evaluation results. Then we discuss the results, perform a model visualization and study the errors produced by our system. Finally, we conclude and discuss future work.

## 2 Related Work

The recently built retrieval-based dialogue systems either match the candidate response with only one dialogue turn of the context "*single-turn*" or with every dialogue turn "*multi-turn*". In the first category, some early studies consider only the last context turn for matching the response [19, 20] or concatenate the context turns and match them with the response [12, 22, 24, 23]. Even if the architecture of these systems is quite simple, some of them require external modules in order to provide topic words or knowledge bases.

On the other hand, the most recent multi-turn systems [21, 25] highlight the importance of matching the response with every context turn. While these systems achieve higher performances, they require more modules (LSTMs, GRUs, CNNs ..) in order to learn representations of every turn in addition to complex matching mechanisms. Thus, the estimation of the number of turns to consider, the training and adaptation of such architectures become a hard task.

In this work, we propose a single-turn<sup>1</sup> response ranking system that matches the candidate response with the context on two levels. Our model is conceptually simpler and can be easily adapted to other domains since it does not require domain related information.

## 3 Multi-Level Retrieval-Based Dialogue System

In this section, we formalize the problem that we address and we describe the architecture of our multi-level retrieval-based dialogue system.

### 3.1 Problem Formalization

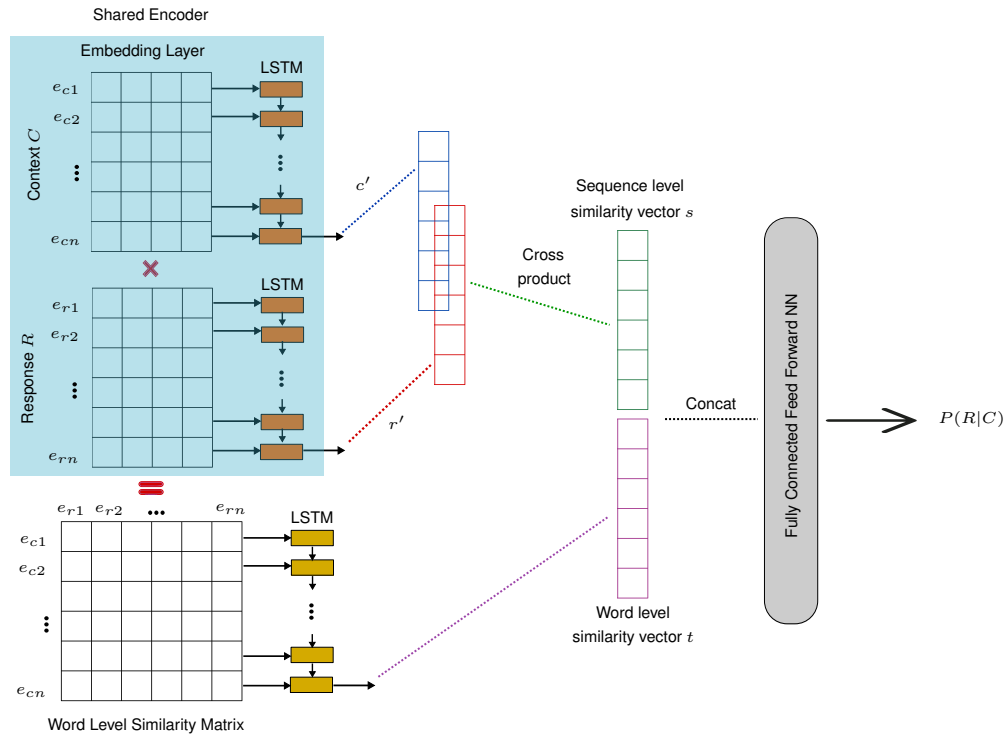
Given a conversation context  $C$  as a succession of  $s$  words  $w_{ci}$  such as  $C = \{w_{c1}, w_{c2}, w_{c3}, \dots, w_{cs}\}$  and a set of candidate responses  $R$  where each candidate response  $R$  is a succession of  $t$  words  $w_{rj}$  such as  $R = \{w_{r1}, w_{r2}, w_{r3}, \dots, w_{rt}\}$ . The problem consists of selecting the best response  $R$  to  $C$ . We define the problem as a ranking task in which we want to order candidate responses by their increasing score of suitability to the conversation context. The utterance with the highest score is then chosen as the next utterance<sup>2</sup>.

### 3.2 System Architecture

We propose an end-to-end multi-level context response matching dialogue system. First, we project the context and the candidate response into a distributed representation (word embeddings).

<sup>1</sup>We concatenate all the context turns as one single context.

<sup>2</sup>Note that throughout this paper we use the terms *next utterance* and *response* indifferently.



**Fig. 2.** Architecture of our multi-level context response matching dialogue system

Second, we encode the context and the candidate response into two fixed-size vectors using a shared recurrent neural network (described in Figure 2 with the blue frame). Then, in parallel, we compute two similarities: on word level and sequence level. The sequence level similarity is obtained by multiplying the context and the response vectors. Whereas the word level similarity is obtained by multiplying word embeddings of the context and the candidate response. Both similarities are concatenated and transformed into a probability of the candidate response being the next utterance of the given context. In the following, we elaborate on the functions of our system.

### 3.2.1 Sequence Encoding

The first layer of our system maps each word of the input into a distributed representation  $\mathbb{R}^d$  by looking up a shared embedding matrix  $E \in \mathbb{R}^{|V| \times d}$  where  $V$  is the vocabulary and  $d$  is the dimension of word embeddings.

We initialize the embedding matrix  $E$  using pretrained vectors (more details are given in 4.4).  $E$  is a parameter of our model to be learned by propagation. This layer produces matrices  $C = [e_{c1}, e_{c2}, \dots, e_{cn}]$  and  $R = [e_{r1}, e_{r2}, \dots, e_{rn}]$  where  $e_{ci}, e_{ri} \in \mathbb{R}^d$  are the embeddings of the  $i$ -th word of the context and the response respectively and  $n$  is a fixed sequence length. Context and response matrices  $C, R \in \mathbb{R}^{d \times n}$  are then fed into a shared LSTM network word by word in order to get encoded.

Let  $c'$  and  $r'$  be the encoded vectors of  $C$  and  $R$ . They are the last hidden vectors of the encoder such as  $c' = h_{c,n}$  and  $r' = h_{r,n}$  where  $h_{c,i}, h_{r,i} \in \mathbb{R}^m$  and  $m$  is the dimension of the hidden layer of the LSTM recurrent network.  $h_{c,i}$  is obtained by Equation 1.  $h_{r,i}$  is obtained similarly by replacing

$e_{ci}$  by  $e_{ri}$ :

$$\begin{aligned} z_i &= \sigma(W_z \cdot [h_{c,i-1}, e_{ci}]), \\ r_i &= \sigma(W_r \cdot [h_{c,i-1}, e_{ci}]), \\ \tilde{h}_{c,i} &= \tanh(W \cdot [r_i * h_{c,i-1}, e_{ci}]), \\ h_{c,i} &= (1 - z_i) * h_{i-1} + z_i * \tilde{h}_{c,i}, \end{aligned} \quad (1)$$

$W_z, W_r$  and  $W$  are parameters,  $z_i$  and  $r_i$  are an update gate and  $h_{c,0} = 0$ .

### 3.2.2 Sequence Level Similarity

We hypothesis that positive responses are semantically similar to the context. Thus, the aim of a response retrieval system is to rank the response that shares the most common semantics with the context on top of the candidate responses. Once the input vectors are encoded, we compute a cross product  $s$  between  $c'$  and  $r'$  as follows:

$$s = c' \wedge r' \equiv s = h_{c,n} \wedge h_{r,n}, \quad (2)$$

where  $\wedge$  denotes the cross product. As a result,  $S \in \mathbb{R}^m$  models the similarity between  $C$  and  $R$  on the sequence level.

### 3.2.3 Word Level Similarity

We believe that sequence level similarity is not enough to match the context with the best response. Adding word level similarity could help the system learning an improved relationship between  $C$  and  $R$ . This assumption was consolidated by observing the scores dropping when word level similarity was removed from our system (see section "Model ablation").

Therefore we compute a word level similarity matrix  $WLSM \in \mathbb{R}^{n \times n}$  by multiplying every word embedding of the context  $e_{ci}$  by every word embedding of the response  $e_{rj}$  as:

$$WLSM_{i,j} = e_{ci} \cdot e_{rj}. \quad (3)$$

In order to transform the word level similarity matrix into a vector, we feed every row  $WLSM_i$  into an LSTM recurrent network which learns a representation of the chronological dependency and the semantic similarity between the context and response words (see Figure 2).

Similarly to Equation 1, we encode the word level similarity matrix into a vector  $T = h'_n \in \mathbb{R}^l$  where  $l$  is the dimension of the hidden layer of the LSTM network and  $h'_n$  is the last hidden vector of the network.

### 3.2.4 Response Score

At this stage we have two vectors:  $S$  representing the similarity between  $C$  and  $R$  on the sequence level and  $T$  representing the word level similarity. We concatenate both vectors and transform the resulting vector into a probability using a one-layer fully-connected feed-forward neural network with sigmoid activation (Equation 4). The last layer predicts the probability  $P(R|C)$  of the response  $R$  being the next utterance of the context  $C$  as:

$$P(R|C) = \text{sigmoid}(W' \cdot (S \oplus T) + b), \quad (4)$$

where  $W'$  and  $b$  are parameters and  $\oplus$  denotes concatenation. We train our model to minimize the binary cross-entropy loss.

The advantages of our system compared to the state of the art ones are: (1) unlike [22] and [20], in our architecture no external module is required to provide extra information such as topic words or related knowledge; (2) we extract sequence and word level similarity with a simple end-to-end architecture that learns to match the context with the best response by considering all the context utterances.

## 4 Experimental Setup

In this section we describe our experimental environment. First we provide a description of the datasets on which we evaluated our system. Then we present the baseline systems and the parameter tuning. Finally we provide the evaluation metrics.

	UDC (V1)			Douban		
	Train	Valid	Test	Train	Valid	Test
# dialogues	1M	500,000	500,000	1M	50,000	10,000
# cand. R per C	2	10	10	2	2	10
Min # turns per C	1	2	1	3	3	3
Max # turns per C	19	19	19	98	91	45
Avg. # turns per C	10.13	10.11	10.11	6.69	6.75	6.47
Avg. # tokens per C	115.0	114.6	115.0	109.8	110.6	117.0
Avg. # tokens per R	21.86	21.89	21.94	13.37	13.35	16.29

**Table 1.** Statistics on the datasets. *C*, *R* and *cand.* denote context, response and candidate respectively

## 4.1 Datasets

**Ubuntu Dialogue Corpus:** [12] collected a large public domain specific corpus of Ubuntu dialogues called the Ubuntu Dialogue Corpus (UDC). The corpus contains conversations with at least three dialogue turns extracted from the chat logs of the channel *#Ubuntu* on the Freenode Internet Relay Chat (IRC)<sup>3</sup>. Conversations from this source are multi users on which heuristics were applied in order to extract two-user discussions. Two versions of this corpus exist. We evaluated our system on the version V1 of the dataset.

Each sample in the training set is a triplet (*context*, *response*, *label*). In the validation and test sets, each sample is made of a context and 10 candidate responses where one is the *ground-truth response* and 9 are negative responses randomly sampled from the corpus. We use the copy shared by [22] in which *numbers*, *urls*, and *paths* were replaced by special placeholders<sup>4</sup>

**Douban Conversation Corpus:** Douban Conversation Corpus<sup>5</sup> is an open domain corpus extracted from Douban Group by [21]. Douban is a public Chinese social network allowing registered users to record information and create content related to film, books, music, recent events and activities in Chinese cities<sup>6</sup>. The corpus contains more than 1 million conversations between two persons with at least three dialogue turns.

<sup>3</sup>For the period 2004-2015 available on <https://irclogs.ubuntu.com/>

<sup>4</sup>Available on [https://www.dropbox.com/s/2fdn26rj6h9bpv1/ubuntu\\_data.zip](https://www.dropbox.com/s/2fdn26rj6h9bpv1/ubuntu_data.zip)

<sup>5</sup>Available on <https://www.dropbox.com/s/90t0qtji9ow20ca/DoubanConversaionCorpus.zip>

<sup>6</sup><https://www.douban.com/group>

Each dialogue sample in the training and validation sets has one positive and one negative responses randomly sampled from the corpus. In the test set, each dialogue sample may have more than one positive response unlike the test set of the Ubuntu Dialogue Corpus. Labelers were recruited in order to judge whether each candidate response is positive or negative (see section 5.2 of [21] for more details about the corpus). We follow [21] and remove test samples with all positive or all negative responses and thus the test set size is reduced to 6,670 samples. According to the authors, Douban Conversation Corpus is the first human-labeled multi-turn response selection dataset. The task on these datasets consists of ranking the ground-truth response on top of the negative responses. Table 1 summarizes statistics on both corpora.

## 4.2 Baselines

We report the results of 7 state of the art systems to which we compare our system. We copy the scores produced by the authors in the original papers.

**TF-IDF** We report results of the Term Frequency-Inverse Document Frequency (TF-IDF) model [13]. The context and each of the candidate responses are represented as vectors of TF-IDF of their words. Then, a cosine similarity is computed between the context and the response vectors and used as a ranking score of the response.

**LSTM dual encoder** The model was introduced in the work of [13]. The context and the response were presented using their word embeddings and then they were fed word by word into two an LSTM network to encode them into fixed size vectors. Then a response ranking score is computed using a bilinear model [15].

**BiLSTM dual encoder** The system of [6] in which the LSTM cells were replaced by bidirectional LSTM cells. We do not report results of their ensemble system which regroups 11 LSTMs, 7 Bi-LSTMs and 10 CNNs because we believe that it is important to build simple systems.

**Deep Learning to Respond (DL2R)** Proposed by [23] based on contextually query reformulation and an aggregation of three similarity scores computed on the sequence level. The reformulated query is matched with the response, the original query and the previous post.

**Multi-View** This system was designed by [24] in which a two similarity levels between the candidate response and the context are computed and the model is trained to minimize two losses. The disagreement loss and the likelihood loss between the prediction of the system and what the system was supposed to predict.

**Sequential Matching Network (SMN)** Proposed by [21]. The candidate response and every dialogue turn of the context are encoded using a GRU network [5]. Then, the response is matched with every turn using a succession of convolutions and max-pooling.

**Deep Attention Matching Network (DAM)** Introduced in the work of [25]. This system is an improvement of the SMN [21] in which the Transformer [17] was used in order to produce utterance representations based on self-attention. These representations are matched together to produce self- and cross-attention scores which are stacked as a 3D matching image. Then, a ranking score is produced from this image via convolution and max pooling operations.

#### 4.3 Evaluation Metrics

The evaluation of conversational systems is an open research domain in which there are no standard evaluation metrics [11, 10]. We followed [12, 20, 22, 21] in using *Recall@k*, *Precision@1*, *Mean Average Precision* (MAP) [1] and *Mean Recall Rank* (MRR) [18] as evaluation metrics. These are common metrics in evaluating IR systems such as recommendation systems and research engines, etc. Note that since in UDC each context has one single positive response in among the candidate responses, we only report

MRR and R@1 as they are equivalent to MAP and P@1 respectively.

#### 4.4 System Parameters

The initial learning rate was set to 0.001 and Adam's parameters  $\beta_1$  and  $\beta_2$  were set to 0.9 and 0.999 respectively. As a regularization strategy we used *early-stopping* and to train the model we used mini batch of size 256. We trained word embeddings of size 300 on UDC and 100 on Douban using FastText [3]. The sizes of the hidden layers of the sequence LSTM and the word LSTM were set to 300 and 200 respectively. The system parameters were updated using Stochastic Gradient Descent with Adam algorithm [7]. All the hyper-parameters were obtained with a grid search on the validation set. We implemented our system with Keras [4] and Theano [16] in backend. We release our source code on [https://github.com/basma-b/multi\\_level\\_chatbot](https://github.com/basma-b/multi_level_chatbot).

### 5 Results and Analysis

In this section we provide a table summarizing the results of our system and the baseline systems in addition to a visualization of the WLSM matrix, an error analysis and a model ablation study.

#### 5.1 Results

Table 2 summarizes evaluation results on UDC (V1) and Douban Conversation Corpus<sup>7</sup>. Compared to the single-turn systems (the first five rows), our system achieves the best results on all metrics and on both datasets. The first four systems are based on only sequence level similarity between the context and the candidate response whereas our system incorporates word level similarity in addition to the sequence similarity. Moreover, our system outperforms the SMN<sub>dynamic</sub> [21] with a good margin (around 4% and 3% on Recall@1 and 2 respectively on UDC). Even if the SMN matches the response with every context turn and uses multiple convolutions and

<sup>7</sup>We limited the number of baseline systems in our table to the most representative ones of each category. For more systems, we refer to the results Table of [21]

System	Ubuntu Dialogue Corpus V1				Douban Conversation Corpus					
	R <sub>2</sub> @1	R <sub>10</sub> @1	R <sub>10</sub> @2	R <sub>10</sub> @5	R <sub>10</sub> @1	R <sub>10</sub> @2	R <sub>10</sub> @5	P@1	MAP	MRR
TF-IDF [13]	0.659	0.410	0.545	0.708	0.096	0.172	0.405	0.180	0.331	0.359
LSTM [13]	0.901	0.638	0.784	0.949	0.187	0.343	0.720	0.320	0.485	0.527
BiLSTM [6]	0.895	0.630	0.780	0.944	0.184	0.330	0.716	0.313	0.479	0.514
DL2R [23]	0.899	0.626	0.783	0.944	0.193	0.342	0.705	0.330	0.488	0.527
Multi-View [24]	0.908	0.662	0.801	0.951	0.202	0.350	0.729	0.342	0.505	0.543
SMN <sub>dynamic</sub> [21]	0.926	0.726	0.847	0.961	0.233	0.396	0.724	0.397	0.529	0.569
DAM [25]	<b>0.938</b>	<b>0.767</b>	<b>0.874</b>	<b>0.969</b>	0.254	0.410	0.757	<b>0.427</b>	<b>0.550</b>	<b>0.601</b>
Our system	0.935	0.763	0.870	0.968	<b>0.255</b>	<b>0.414</b>	<b>0.758</b>	0.418	0.548	0.594
Only sequence similarity	0.917	0.685	0.825	0.957	0.209	0.357	0.702	0.358	0.500	0.543
Only word similarity	0.926	0.744	0.853	0.956	0.223	0.370	0.719	0.373	0.513	0.556

**Table 2.** Evaluation results on the UDC V1 and Douban Corpus using retrieval metrics

max pooling to rank the response, its performance is lower than our system's performance. We believe that using our architecture, we were able to efficiently capture both similarity levels.

Our system neither matches each context turn with the candidate response nor uses complex cross and self attention in addition to matching and accumulation mechanisms but achieves almost the same performance as the Deep Attention Matching (DAM) [25] on both datasets and on all metrics. The DAM as detailed in Section 4.2 is based on multiple layers of the self attention (Transformer) and Convolutional Neural Networks [8]. Even if the advantages of the Transformer are related to the performance improvement and the acceleration of the learning compared to neural networks [17]. However, we proposed an architecture that is fully based on neural networks but that achieves almost the same results as the DAM and sometimes better. The advantages of our system compared to the DAM is in contrast to what was said before, our system converges quickly. According to the authors [25], their system was trained on one Nvidia Tesla P40 GPU, on which one epoch lasts for 8 hours on UDC and their system converges after 3 epochs.

However, training our system for one epoch lasts for 50 minutes on one Nvidia Titan X pascal GPU (Both GPUs have almost the same characteristics<sup>8</sup>) and our system converges after two epochs<sup>9</sup>. Having such architectures (as

DAM) makes reproducibility of results harder due to hardware limitations and time necessary to perform training and cross-validation.

Note that on Douban, the overall performance of all the systems are lower than on UDC. This is due to the nature of Douban corpus in which a context may have more than one ground-truth response and hence every retrieval system must find all the responses.

## 5.2 Error Analysis

We performed a human evaluation of 200 randomly selected test samples from UDC where the ground-truth response was not retrieved by our system. By observing the test samples that were misclassified, we identified 4 error classes. Table 3 summarizes the distribution of the test samples over these classes. Around 50% of the errors are cases where our system produced a response that is either functionally or semantically equivalent to the ground-truth response.

Error class	Percentage
Functionally equivalent	31%
Semantically equivalent	20%
Out of context	35.5%
Very general responses	13.5%

**Table 3.** Error classes

In fact, considering these cases as errors may falsify the evaluation. Surprisingly, the other half of errors are due to out of context and very general responses. This drawback was usually noticed in

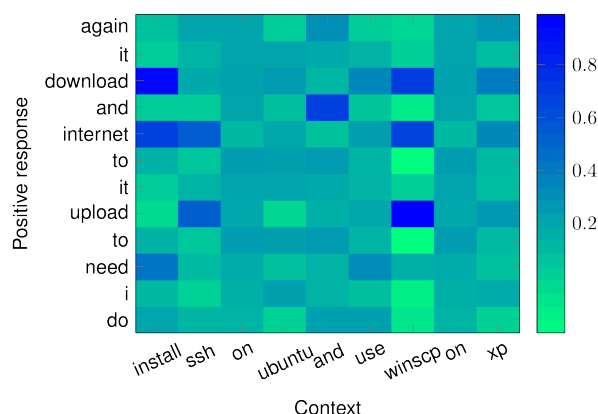
<sup>8</sup><https://technical.city/en/video/Titan-X-Pascal-vs-Tesla-P40>

<sup>9</sup>The number of trainable parameters of our system and DAM is almost the same

generative dialogue systems, however, in this case of study, it is also a major drawback of our retrieval-based dialogue system.

These findings encourage us to perform a deep comparative study between these two categories of dialogue systems.

### 5.3 Visualization



**Fig. 3.** Visualization of the Word-Level Similarity Matrix (WLSM)

Furthermore, we visualized WLSM for the following test sample. The last turn of the context is:

**A:** *hey anybody know how i can share file between xp guest and ubuntu 12.04 lts host in vmware ?*

**B:** *"install ssh on ubuntu and use winscp on xp".*  
The positive response is *"do i need to upload it to internet and download it again"*.

In Figure 3, we plotted the Word Level Similarity Matrix WLSM between the context (x-axis) and the response (y-axis). For a matter of space we visualize only the last dialogue turn (**B**) of the context. As we can see, important (key) words in the context and the response were successfully recognized by our system and were given higher scores.

For instance, *upload*, *internet* and *download* were matched with *install*, *ssh*, *winscp* and *xp*. This observation illustrates the importance of computing word level similarity from word embeddings in order to match the context with the best response.

### 5.4 Model Ablation

We report in the two last rows of Table 2 the performance of our system while having only one similarity level. We notice that having only one level of similarity causes a drop of the system performance. Results are higher when matching the context with the candidate response on the word level compared to the sequence level. Considering the example of Section 5.3, the whole context and the response are semantically similar. Having in addition to this sequence similarity, the fact that *upload*, *internet* and *download* match with *install*, *ssh* and *winscp* will help the system better recognizing the good responses. Vice versa, we can have responses that share semantically equivalent words with the context while the whole meaning of the response is not related to the whole meaning of the context.

These results highlight the importance of considering both similarity levels in our system in order to achieve higher performances. Note that there is a slight difference in the performance of our system with only one similarity level on both datasets. We believe that this is related to the characteristics of each corpus.

## 6 Conclusion

We presented a simple and efficient multi-level retrieval-based dialogue system. Our system learns to match the context with the best response based on their similarity that we capture on word and sequence levels with a simple architecture. By learning a word level and sequence level similarities our system was able to capture deep relationships between the context and the candidate responses. The experimental results on two large datasets demonstrate the efficiency of our approach by bringing significant improvements compared to complex state-of-the-art systems.

In essence, a simple model can suffice to achieve good performance, sometimes even better than complex response matching models. As future work, we will extend this study by investigating the possibility of adding more similarity levels while keeping the simplicity of the architecture. Moreover, we plan to enrich text with



discursive information such as dialogue acts and rhetorical relations.

## 7 Acknowledgment

We thank the anonymous reviewers for their valuable comments. This work was partially supported by the project ANR 2016 PASTEL<sup>10</sup>.

## References

1. Baeza-Yates, R. A. & Ribeiro-Neto, B. (1999). *Modern Information Retrieval*. Addison-Wesley Longman Publishing Co., Inc., Boston, MA, USA.
2. Baudiš, P., Pichl, J., Vyskočil, T., & Šedivý, J. (2016). Sentence pair scoring: Towards unified framework for text comprehension. *arXiv preprint arXiv:1603.06127*.
3. Bojanowski, P., Grave, E., Joulin, A., & Mikolov, T. (2017). Enriching word vectors with subword information. *Transactions of the Association of Computational Linguistics (TACL)*, Vol. 5, pp. 135–146.
4. Chollet, F. et al. (2015). Keras. <https://github.com/keras-team/keras>.
5. Chung, J., Gulcehre, C., Cho, K., & Bengio, Y. (2014). Empirical evaluation of gated recurrent neural networks on sequence modeling. *Workshop on Deep Learning and Representation Learning at the 28th Annual conference on Advances in Neural Information Processing Systems (NIPS'14)*, Montreal, Canada.
6. Kadlec, R., Schmid, M., & Kleindienst, J. (2015). Improved deep learning baselines for ubuntu corpus dialogs. *Workshop on Machine Learning for Spoken Language Understanding and Interaction at the 29th Annual Conference on Neural Information Processing Systems (NIPS'15)*, Montreal, Canada.
7. Kingma, D. & Ba, J. (2015). Adam: A method for stochastic optimization. *Proceedings of the 3rd International Conference for Learning Representations (ICLR'15)*, San Diego, CA, USA.
8. LeCun, Y., Bottou, L., Bengio, Y., & Haffner, P. (1998). Gradient-based learning applied to document recognition. *Proceedings of the IEEE*, Vol. 86, No. 11, pp. 2278–2324.
9. Li, J., Galley, M., Brockett, C., Gao, J., & Dolan, B. (2016). A diversity-promoting objective function for neural conversation models. *Proceedings of the 2016 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies (NAACL'16)*, San Diego, CA, USA, pp. 110–119.
10. Liu, C.-W., Lowe, R., Serban, I., Noseworthy, M., Charlin, L., & Pineau, J. (2016). How not to evaluate your dialogue system: An empirical study of unsupervised evaluation metrics for dialogue response generation. *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing (EMNLP'16)*, Austin, Texas, pp. 2122–2132.
11. Lowe, R., Noseworthy, M., Serban, I. V., Angelard-Gontier, N., Bengio, Y., & Pineau, J. (2017). Towards an automatic turing test: Learning to evaluate dialogue responses. *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (ACL'17)*, Vancouver, Canada, pp. 1116–1126.
12. Lowe, R., Pow, N., Serban, I., & Pineau, J. (2015). The ubuntu dialogue corpus: A large dataset for research in unstructured multi-turn dialogue systems. *Proceedings of the 16th Annual Meeting of the Special Interest Group on Discourse and Dialogue (SIGDIAL'15)*, Prague, Czech Republic, pp. 285–294.
13. Lowe, R. T., Pow, N., Serban, I. V., Charlin, L., Liu, C.-W., & Pineau, J. (2017). Training end-to-end dialogue systems with the ubuntu dialogue corpus. *Dialogue & Discourse*, Vol. 8, No. 1, pp. 31–65.
14. Shao, Y., Gouws, S., Britz, D., Goldie, A., Strophe, B., & Kurzweil, R. (2017). Generating high-quality and informative conversation responses with sequence-to-sequence models. *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing (EMNLP'17)*, Copenhagen, Denmark, pp. 2210–2219.
15. Tenenbaum, J. B. & Freeman, W. T. (2000). Separating style and content with bilinear models. *Neural computation*, Vol. 12, No. 6, pp. 1247–1283.
16. Theano Development Team (2016). Theano: A Python framework for fast computation of mathematical expressions. *arXiv e-prints*, Vol. abs/1605.02688.
17. Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A. N., Kaiser, Ł., & Polosukhin, I. (2017). Attention is all you

<sup>10</sup><http://www.agence-nationale-recherche.fr/?Projet=ANR-16-CE33-0007>

- need. *Advances in Neural Information Processing Systems (NIPS'17)*, Long Beach, CA, USA, pp. 5998–6008.
18. Voorhees, E. M. (2001). The trec question answering track. *Natural Language Engineering*, Vol. 7, No. 4, pp. 361–378.
  19. Wang, H., Lu, Z., Li, H., & Chen, E. (2013). A dataset for research on short-text conversations. *Proceedings of the Conference on Empirical Methods in Natural Language Processing (EMNLP'13)*, Seattle, WA, USA, pp. 935–945.
  20. Wu, Y., Wu, W., Li, Z., & Zhou, M. (2016). Response selection with topic clues for retrieval-based chatbots. *arXiv preprint arXiv:1605.00090*.
  21. Wu, Y., Wu, W., Xing, C., Zhou, M., & Li, Z. (2017). Sequential matching network: A new architecture for multi-turn response selection in retrieval-based chatbots. *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (ACL'17)*, Vancouver, Canada, pp. 496–505.
  22. Xu, Z., Liu, B., Wang, B., Sun, C., & Wang, X. (2017). Incorporating loose-structured knowledge into conversation modeling via recall-gate lstm. *Proceedings of the International Joint Conference on Neural Networks (IJCNN'17)*, Anchorage, AK, USA, pp. 3506–3513.
  23. Yan, R., Song, Y., & Wu, H. (2016). Learning to respond with deep neural networks for retrieval-based human-computer conversation system. *Proceedings of the 39th International ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR '16)*, Pisa, Italy, pp. 55–64.
  24. Zhou, X., Dong, D., Wu, H., Zhao, S., Yu, D., Tian, H., Liu, X., & Yan, R. (2016). Multi-view response selection for human-computer conversation. *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing (EMNLP'16)*, Austin, Texas, pp. 372–381.
  25. Zhou, X., Li, L., Dong, D., Liu, Y., Chen, Y., Zhao, W. X., Yu, D., & Wu, H. (2018). Multi-turn response selection for chatbots with deep attention matching network. *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (ACL'18)*, Melbourne, Australia, pp. 1118–1127.

Article received on 17/01/2019; accepted on 04/03/2019.  
Corresponding author is Basma El Amel Boussaha.

# A New Approach for Twitter Event Summarization Based on Sentence Identification and Partial Textual Entailment

Dwijen Rudrapal<sup>1</sup>, Amitava Das<sup>2</sup>, Baby Bhattacharya<sup>1</sup>

<sup>1</sup> National Institute of Technology, Agartala, India

<sup>2</sup> Wipro AI Lab, Bangalore, India

dwijen.rudrapal@gmail.com, amitava.santu@gmail.com, babybhatt75@gmail.com

**Abstract.** Recent trend of information propagation on any real-time event in Twitter makes this platform more and more popular than any other online communication media. This trend creates a necessity to understand real-time events quickly and precisely by summarizing all the relevant tweets. In this paper, we propose a two-phase summarization approach to produce abstract summary of any Twitter event. The approach first extracts key sentences from the whole set of event relevant tweets and eliminates maximum redundant information by exploring Partial Textual Entailment (PTE) relation between sentences. Next, generates an abstract summary over the least redundant key sentences. We conduct experiments to evaluate the performance of our propose approach and report that the approach outperforms over the baseline approach as well as state-of-the-art event summarization approach.

**Keywords.** Social media text, twitter event, summarization, partial textual entailment, tweet ranking.

## 1 Introduction

Summarization is a process of presenting an event's most important and relevant information in precise way. It helps to understand an event very quickly and precisely. The significance of summary becomes more imperative and effective while dealing with events in social media platforms like Twitter [34] where information is fast, rich content, diverse and ever increasing. Irrespective of any geographic location, time and other conditions, any user can post comments or views, upload videos on any event very quickly. Thus, Twitter has

become one of the most popular online information sharing platforms and source of breaking news as well. Twitter is the first to report information on many events like earthquake in North East India, terrorist attacks in German and France, missing of m327 flight, status on USA President election, status and result of sports events and many more before any traditional news media. However, the information shared in Twitter for an event often leads to the information overload problem [21, 45] leaving very less number of event relevant posts [53]. Though, the summarization task of Twitter events received a lot of attention from research world throughout the last decade due to the sharing of fast and diverse response from millions of users.

In general, summarization is a user specific task where generated summaries are varied due to the diverse views of the users concerning the event. While, summarization of Twitter event is more difficult than traditional documents due to the different nature of text genre which poses several challenges. Such as:

1. **Processing of tweet content:** Tweets are limited to 280 characters and often published without proofreading. Often, tweets content include spelling mistakes, grammatical errors, self created acronyms, out-of-vocabulary words and very short comments from online discussion of users. In addition, a large number of tweets in an event are irrelevant to that event or relevant to other event. Thus, standard Natural Language Processing (NLP) tools do not work well on this genre of text and

and creates challenges for processing of tweet contents, clustering and topic modeling task due to the high perceptive nature of features.

2. **Sentence boundary detection (SBD):** A tweet is not always a single sentence, rather, it may include multiple sentences. Sentences are essential prerequisite for any summarization task. But, SBD in tweets is very challenging task [36] due to the use of punctuation in a nonstandard manner. Sometimes, tweet includes no punctuation at all even though it includes multiple sentences. For example, first tweet of following examples includes two sentences with nonstandard punctuation as sentence end marker. Second tweet includes 3 sentences while 2nd and 3rd sentences have no any sentence end marker.

***Tweet 01:*** *#USA + #Europe + #Asia must help #Africa ... #Liberia cannot cope alone with #Ebolaoutbreak <http://t.co/IXdRW4Q3BB>*

***Tweet 02:*** *Modiji bi-election results show that people of Bihar and UP can not break the caste-religious bonds . They r poor they do't prefer growth*

3. **Information redundancy problem:** Events in Twitter attract large volume of tweets which makes difficult to identify the most important tweets for human too. Every event contains redundant information for an event in the form of Re-tweet or tweet having fully or partially entailed contents of another tweet at high volume. In order to reduce redundant information, identification of partially entailed information among tweets is a more difficult task than identifying re-tweet or entailed tweets. For example, two sentences are cited below from two different tweets. Second sentence contains entailed information from first sentence and additional content (highlighted). In order to reduce redundant information, these two sentences can be merged into one meaningful sentence as shown below.

***Sentence 01:*** *Its #Jallikattu day at the world famous #Alanganallur.*

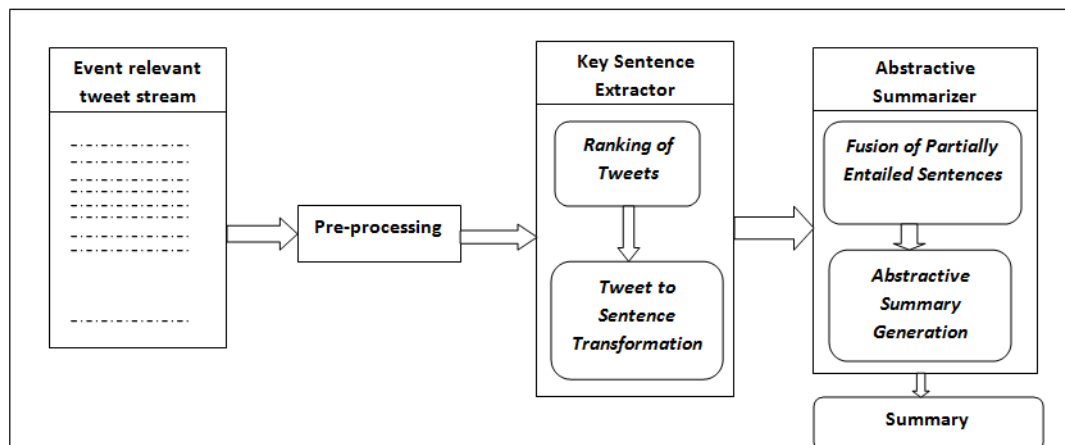
***Sentence 02:*** *World Famous #Alanganallur #Jallikattu back after 3 years.*

***Merged Sentence:*** *Its #Jallikattu day at the world famous #Alanganallur back after 3 years.*

This paper proposes a summarization approach to generate an abstractive summary of an event from Twitter by providing potential solution to above challenges. Figure 1 presents the structure of our summarization approach. Abstractive summarization [26, 23] characterizes an event in more compact way with the cost of high time-complexity. Our approach follows extractive-abstractive structure to generate summary over most informative and relevant sentences from entire tweet stream of an event. Experimental results in section 4 confirm that the proposed summarization approach performs better than state-of-the-art summarization approach. Our main contributions in this paper are:

1. In most of the earlier research work, a tweet is considered as a sentence for summarization task. But in reality, often tweets include multiple sentences. In this research work we extract key sentences by SBD in tweets rather than treating a tweet as a sentence. In order to extract most important sentences for an event, we ranked all the event relevant tweets first and then transformed top ranked tweets into possible sentences.
2. In order to generate abstract summary, we filter out redundant information by exploring Partial Textual Entailment (PTE) relationship among sentences. We recognize partially entailed (PE) sentences and the boundary of partially entailed text among sentences. A pair of PE sentences merged into single sentence by combining PE text. Finally, an abstract summary generates over selective sentences by covering maximum information, diversity, coherence and readability.

This paper is structured as follows. Section 2 presents current research progress of Twitter event summarization. Section 3 presents the proposed summarization approach followed by experimental



**Fig. 1.** Structure of Proposed summarization approach

setup and results in section 4. Section 5 reports analysis of result and finally, section 6 concludes this paper.

## 2 Related Work

Our proposed approach follows extractive-abstractive structure to generate summary of an event. Therefore, we review promising summarization works developed based on extractive and/or abstractive approaches.

Extractive summary of an event is the set of most relevant and informative tweets in the event. Based on desired size, summary includes number of most relevant and informative tweets. In 2010, Sharifi et al. [40] introduced a “Phrase Reinforcement (PR) Algorithm” to generate summary of one tweet length for any Twitter event. The Algorithm constructs an ordered acyclic graph with the relevant tweets as nodes and computes weight of each node from the frequency of occurrences. Maximum total weight path in the graph is generated as final summary. The work further extended to maximize the coverage of the event with more number of tweets [41] and to increase summary readability [18].

Another approach [49] constructs a weighted graph with bag of bi-grams as node and rank each node using TextRank algorithm [22] to finds top ranked bi-grams for summary generation.

Harabagiu and Hickl proposed work [13] produces a summary in 250 words rather than tweets to cover more information about an event. Important moment of any event attracts longer conversation which identifies significant and interesting tweets to produce summary [29]. Tweet text has some specific features like tweet relevance closeness to initial text, relevance regarding URL is used to measure social influence of tweets for summary generation [2]. The work by [25] explored more static social features like re-tweet information, user influence and temporal information to designate a tweet's importance and relevance to be included in summary of an event.

The work proposed in [20] collects tweets and identifies event based on the event specific activities like temporal, spatial and user behavior of that event. Then relevant tweets are ranked to generate summary. Tweet-level social information relation [14] and participant-centered social information [15] are used in recent summarization research work for generating more robust summaries. In another recent work, Chellal et al [5] formulated tweet summarization problem as optimization problem and generate summary as a subset of tweets those are more relevant, diverse and cover maximum information of the event.

Above approaches generate summaries considering all the relevant tweets as a set. But in Twitter, flow of tweets for any event varies

with the progress of time since its inception. Important moment of an event attracts more important and a good number of tweets though those tweets share similar kind of information. Due to this nature of event life in Twitter, various summarization approaches follow clustering method to summarize event. Clustering process divide an event into multiple segments and extracts most important tweets from each segment.

The work by Inouye [16] first introduced multiple cluster based event summary. But, the clustering task is challenging for Twitter due to that the cluster numbers are varies with topic. Thus clustering process highly affects summarization approaches and also needs optimization. Tweet stream clustering process follows various criteria like time window [48, 44], burstiness of tweets [53, 42, 7, 51], semantic closeness [4] and many more. Multiple approach of clustering like stream-based and semantic-based approaches are also used to clusters relevant tweets into subtopics [9] for summary generation. The work proposed in [52] used an online incremental clustering algorithm to form sub-events. Tweets in each cluster ranked based on the features like noun phrases, verbs, hashtags, URLs and numbers. Top ranked tweet from each cluster constructs the summary of the event.

An abstractive summary presents an event with key information rather than citing source tweets. With limited number of words, the abstractive summary covers more information about an event than extractive summary. The work proposed in [26] produces summary based on word graph and optimization techniques. The graph constructs with word tri-grams and weighted based on the occurrence frequencies. Rudra et al. [32] produces abstract summary for domain specific Twitter events like crisis scenario events by constructing a graph with word bigrams and word co-occurrences relation. The work [50] represents content of tweet in 5 types of speech act based on Searle's taxonomy of speech acts [37]. The algorithm extracts word and symbol based features for each tweet and labels the tweet by corresponding speech act using Support Vector Machine (SVM) classifier. Topic words and the salient words/phrases for

each major speech act type are selected by using round robin algorithm and insert into proper slots of speech act-guided templates to generate abstractive summary. Shapira et al. introduced approach [39] extracts facts of the event and expands information of the facts through alternative terms to generate summary for news event tweets.

All the previous works assume a tweet as a sentence. But in actual, a tweet includes multiple sentences in it where all of them are not contain equally important information. Earlier works eliminated redundant information by exploring semantically similar sentences only. We explored and identify partially entailed text between sentences to eliminate maximum redundant information. In this research work, we generate summaries over key sentences rather than key tweets and eliminate maximum redundant information in sentences by exploring partially entailed information.

### 3 Proposed Summarization Approach

Our proposed approach of summarization consists with two components: (1) "Key sentence extractor", which extracts most relevant and informative sentences rather than tweets from the set of event relevant tweet stream. (2) "Abstract summarizer", which generates abstractive summary over selected sentences after removing redundant information by identifying entailed or partially entailed information in sentences. Figure 1 provides an overview of the summarization approach.

The components of our summarization approach are described below.

#### 3.1 Key Sentence Extractor

Sentence identification in tweets is a very difficult and ambiguous task due to the grammatical structure of the content and inappropriate practice of punctuation. Thus, key sentence extractor component first ranks event relevant tweets based on relevance and informativeness. Next, top ranked tweets are converted into possible sentences. Finally, most relevant and informative sentences are extracted as key sentences. In the

following subsections, we elaborate the procedure of key sentence extraction.

### 3.1.1 Ranking of Tweets

A Twitter event  $E$  includes sequence of relevant tweets  $(t_1, t_2, t_3, \dots, t_n)$ . However, all the tweets are not equally important or even necessary to understand or know the event precisely. Every tweet has its own credibility to express event related information. Based on the credibility and significance, learning to rank algorithm distinguish each tweet and rank accordingly to identify most relevant and informative tweets. We follow a pair wise ranking approach in this work to list down most informative and relevant tweets for an event. We formulate the ranking task as classification problem and assign precedence to one tweet over other. For each tweet  $t_i \in (t_1, t_2, t_3, \dots, t_n)$ , algorithm assigns a rank  $y_m \in 1, \dots, n$  for a given Twitter event. Learning to rank algorithm trains a function  $h$  to measure the credibility of a tweet  $h(t)$  so that for any pair of tweets  $(t_i, y_i)$  and  $(t_j, y_j)$ ,

$$h(t_i) > h(t_j) \Leftrightarrow y_i > y_j. \quad (1)$$

We ranked tweets of an event based on the relevance and informativeness of tweet content. Relevance of a tweet to an event is the content's meaning closeness to the event and informativeness is the content's information richness to understand the event. Different features to determine the most important tweet of an event have been studied and employed in earlier works [3, 12, 11, 47]. Our ranking task draw some of the important features like length, unique words, Hashtag, Re-tweet, URL, User mention and user account features like follower, Like, List score from those previous promising tweet ranking approaches. In addition of these standard features, we introduce following 4 new features to measure relevance and informativeness of a tweet.

1. **Distribution of event keywords:** Every event has a list of words while few of them occur most frequently and very important to the event. Distribution of those most frequent words in a tweet represents its relevance to the event more closely.

2. **Semantically equivalent tweet count:**

Semantically similar content can be shared through different expressions without re-tweeting a tweet. So, number of semantically equivalent tweets of a tweet is used to measure the popularity of the content.

3. **Ratio of unique words:** More number of unique words than Twitter specific words represents more rich information. This feature represents ratio of unique words with Twitter specific words in a tweet.

4. **Presence of event top hashtag:** Most frequent hashtag of an event is the most relevant hashtag. So, presence of event's top hashtag in a tweet make it more relevant to the event.

Based on above features and RankSVM algorithm proposed by Joachims [17], we propose a ranking model to rank event relevant tweets. In order to extract key sentences, we select top ranked 20 tweets for each event where number of relevant tweets are always more than 20 tweets.

### 3.1.2 Tweet to Sentence Transformation

Sentences are basic units for any summarization tool. In traditional text like news articles, sentences are easily identified due to the proper use of punctuations. But, in social media platform like Twitter, sentence boundary detection (SBD) is a difficult task. Punctuations are applied in creative and non-standard way which creates several challenges for SBD in tweets. Often a tweet includes multiple sentences in it. Using the SBD tool for social media text, developed in the work [36], we transformed each ranked tweet into possible sentences. The tool has transformed most of the tweets into possible sentences successfully. For example,

**Tweet:** Here we go #Palamedu #Alanganallur Jallikattu here ON .. We tried our best to make big with the help of @iamsridhu #Studiocr ...

**Sentence 01:** Here we go #Palamedu #Alanganallur Jallikattu here ON ..

**Sentence 02:** *We tried our best to make big with the help of @iamsridhu #Studiocr ...*

The accuracy of the SBD tool for our corpus is 84.9%. Few of the tweets where use of punctuation is ambiguous due to emphasis for multiple sentences, tool could not transformed them into possible sentences. Those tweet transformation is done through manual process.

Tweet transformation sometimes produces sentences which are not meaningful or contains only Twitter specific keywords. Thus, we follow one processing step to filter out such sentences satisfying one of the following conditions as they do not hold any key information about the event: (1) Sentence with word length less than 3, (2) Sentence containing only Hashtags, user mentions and URLs, (3) Contains only topic words.

After filtering process, we are left with selected key sentences which are most relevant and informative. Detailed statistics are reported in Table 1.

**Table 1.** Statistics of the corpus

Event Count	Ranked Tweet Count	Sentence Count	Max/Min Sentence/Event
25	500	636	36/11

### 3.2 Abstract Summarizer

In this phase, we analyze key sentences to find and remove maximum redundant information from selected key sentences.

#### 3.2.1 Fusion of Partially Entailed Sentences

Event-focused sentences often include redundant information due to the fact that similar kind of information shared through different expressions [33]. An expression may include semantically similar or partially similar content of another expressions. We have explored redundant information in sentences through identifying PTE relation between them as proposed in the work [6, 35]. We also identify the boundary of the information in one sentence that is partially entailed (PE) to the content of other sentence. This PE information among sentences is combined

to merge sentences into single sentence. In the following subsections we explain PTE identification task and the detection of PE text to merge sentences.

PTE [6, 35] is a bidirectional relationship between two expressions where one expression is entailed or partially entailed from another expression. PTE extends the concept of Textual Entailment (TE). With preservation of formal entailment definition, different categories of PTE are defined by breaking down both the expressions. This categories are PTE-I, PTE-II, PTE-III and PTE-IV. First 3 categories extends true entailment relation and last category is same as negative entailment relation. In this current work, we have generated a PTE recognition model for English tweets following the approach proposed in the work [35].

**Identification of PTE:** Our entailment decision criterion is based on various similarity scores calculated on pair of sentence. Each pair of sentence ( $s_1, s_2$ ) is represented by a feature vector, where each feature is a specific similarity score represents whether  $s_1$  or segment of  $s_1$  entails  $s_2$  or segment of  $s_2$ . We have computed 8 similarity scores for each pair of sentence. The first 4 scores are computed based on the exact lexical overlap between the words like unigram, bi-gram, longest common sub-sequence and skip-gram similarity score. The other similarity scores are cosine similarity, semantic similarity, word-to-vector similarity and soft cosine similarity [43] scores. First three features have been evaluated as effective for PTE class prediction [35] and are briefly explained below. The last feature called soft cosine similarity [43] is calculated based on the similarity of word features in Vector Space Model (VSM). This measure represents closeness of two texts more precisely by ignoring variations of words in texts.

1. **Cosine Similarity:** Cosine similarity score represents the relatedness between two n-dimensional vectors. We converted two sentences into binary vectors with values either 0 or 1 and calculated similarity score



using the following equation:

$$\cos(\vec{A}, \vec{B}) = \frac{\vec{A} \cdot \vec{B}}{\|\vec{A}\| \cdot \|\vec{B}\|}. \quad (2)$$

**2. Semantic Similarity:** We modeled the semantic similarity of two sentences as a function of the semantic similarity of the constituent words in both. Similarity score is calculated by using text to text similarity measure proposed in the work [28].

**3. Word-to-vector Similarity:** To measure word-to-vector similarity of a word, we used deep-learning library Word2Vec toolkit<sup>1</sup>. Word2Vec tool returns similarity score of two words based on a pre-trained word embedding for English tweets [27]. Word level similarity score accumulated to find sentence level similarity using following equation:

$$w2v_{sim}(A, B) = \frac{\sum_{i=0}^{i=N} \max_{sim}(w_i)}{N}, \quad (3)$$

where  $\max_{sim}(w_i)$  is the maximum similarity score of a word in a sentence and N is the total number of unique words in sentence pair.

Feature profile is generated for all sentence pairs from the similarity scores to train and generate PTE identification model. We have selected a subset of the Twitter Paraphrase Corpus as in [46] for training and PTE identification model generation. This dataset is more similar to our PTE identification task and have been studied extensively to predict semantic equivalence between sentences for many NLP applications including summarization. From the chosen subset of the corpus, we manually annotate 1100 pair of sentence into four types of PTE pairs to prepare training data. Manual annotation process employed two human annotators who are post graduate students and native English speaker to annotate each pair of sentence with a category as mentioned in the work [35]. Each annotator annotates every pair of sentence to a class of PTE. Based on the similarity scores of each pair and optimization of

their relative weights, we train a Support Vector Machine (SVM) with 10-fold cross validation to generate PTE identification model. This model is applied on event-focused key sentence pairs to find PTE relation. We form sentence pairs for an event by comparing one sentence with all other sentences in the same event.

**Detection of Partially Entailed Text Boundary:** After identifying PTE sentence pairs, we merge each pair ( $s_1$  and  $s_2$ ) into a single sentence based on following cases:

1. If a pair of sentence is identified as PTE-I, then one sentence should be eliminated;
2. If a pair of sentence is identified as PTE-II, then the sentence having more information in addition of the entailed or entail information will remain in the list and other one will be eliminated.
3. If a pair of sentence is identified as PTE-III, then generate an understandable informative sentence that maximally captures the content of the both sentences.
4. If a pair of sentence is identified as PTE-IV then both the sentences should remain under consideration for further process.

From the above facts, case 1 and case 4 are straightforward where one/both sentence will remain. But case 2 and 3, where segment of a sentence is entails or entailed to other sentence or segment of sentence, boundary of the entailed text is to be identified for merging sentences. To detect the boundary of PE information, each sentence is divided into fragments i.e. piece of information. A piece of information means collection of consecutive words grouped together. A given sentence (s) can be represented as a set of its fragments like:

$S_i = F_j, F_{j+1}, F_{j+2}, \dots, F_m$ , where  $i, j = 1$  to any finite integer.

Using Ritter's Twitter tool [30, 31], we identify phrases in a sentence and consider a phrase as a fragment. In following examples we cited 2 pairs of sentences for PTE-II and PTE-III category respectively.

**PTE-II:**

<sup>1</sup><http://deeplearning4j.org/word2vec.html>

**Sentence 01:** [Its #Jallikattu day]/NP [at]/PP [the world famous #Alanganallur]/NP [.] /O

**Sentence 02:** [World Famous #Alanganallur]/NP [#Jallikattu]/HT [back]/NP [after 3 years]/NP [.] /O

PTE-III:

**Sentence 01:** [Japan]/NP [to invest]/VP [\$35 billion]/NP [in]/PP [India]/NP [over]/PP [the next 5 years]/NP

**Sentence 02:** [Investment worth]/NP [\$35 billion]/NP [to finance]/VP [infrastructure projects]/NP [and]/O smart cities]/NP [in]/PP [India]/NP [in]/PP [the next five years]/NP

We compared each fragment of a sentence with all the fragments in other sentence to identify semantically equivalent and non-equivalent fragments in the pair. Semantically equivalent fragments are recognized based on their semantic closeness [28]. Based on the semantic equivalent fragments, two sentences are merged into one as shown in figure 2 and 3. Resultant sentence is generated from partially entailed and non-entailed information.

### 3.2.2 Summary Generation

In this subsection, we explain abstract summary generation process over selected most relevant and informative sentences. In order to cover maximum information about an event in limited number of words, we generate abstract summary following the work presented in [38]. Proposed approach uses pointer generator network with coverage technique to avoid unknown words and remove repetition of words in final summary. This approach generates words for final summary from the fixed size vocabulary or source text and thus resolve unknown words generation problem in the result summary. Due to the similar attention to a particular piece of text, sometimes repetitions of words occur in summary. Coverage technique in the proposed work uses attention distribution to track attention of a word it has received at any point. The technique applies a loss term to penalize next attention to the same word and thus resolve word repetition issue. Thus, we have chosen pointer-generator network with coverage

technique for our abstract summary generation task.

## 4 Experiment Setup and Result

In this section, we present our experimental setup for assessing the performance of our proposed summarization approach. We describe our dataset, experiment setup, evaluation metrics and comparative performance.

### 4.1 Dataset

We have collected English tweets for 25 trending events using Twitter4j<sup>2</sup> during the period from January to October, 2017. Events are current happenings like natural disaster, politics, sports, entertainment and technology. In order to extract relevant tweets, we used commonly known hashtags and combination of keywords associated with those events. We tokenize each tweet using CMU tokenizer [10]. Our queries return English as well as Non-English tweets containing query hashtags and keywords. So, at initial level we cleaned all the obtained tweets by filtering out Non-English, spam and short tweets with less than 3 words. Statistics of this dataset is shown in Table 2.

**Table 2.** Statistics of the corpus

Event Nos	Tweet Nos	Filtered Tweet	Token nos
25	64,436	14,135	2,24,934

For our experiment and evaluation work, we generate gold summaries for each event consists with 100 tokens. This summary generation task is carried out by three PG students who are native English speaker. We have supplied each event relevant tweets and brief information about each event to each annotator and asked to write abstract summaries of 100 token for each event. Since, generation of gold summary for any event is a difficult task due to the diverse understanding of the event, we obtain Inter Annotator Agreement (IAA) scores as ROUGE values to evaluate the quality of

<sup>2</sup><https://github.com/Twitter4J/Twitter4J>

Sentence 01:

Phrase	Its #Jallikattu day	at	<i>the world famous</i> <i>#Alanganallur</i>
Fragments	F <sub>1</sub>	F <sub>2</sub>	F <sub>3</sub>

Sentence 02:

Phrase	<i>World Famous</i> <i>#Alanganallur</i>	#Jallikattu	back after 3 years
Fragments	F <sub>1</sub>	F <sub>2</sub>	F <sub>3</sub>

**Resultant sentence:** *Its #Jallikattu day at the world famous*  
*#Alanganallur back after 3 years*

**Fig. 2.** Fusion of PTE-II category sentence pair

Sentence 01:

Phrase	Japan	to invest	\$35 billion	in	India	over	<i>the next 5</i> <i>years</i>
Fragments	F <sub>1</sub>	F <sub>2</sub>	F <sub>3</sub>	F <sub>4</sub>	F <sub>5</sub>	F <sub>6</sub>	F <sub>7</sub>

Sentence 02:

Phrase	Investment worth	\$35 billion	to finance	infrastructure projects	and	smart cities	in	India	In	<i>the</i> <i>next</i> <i>five</i> <i>years</i>
Fragments	F <sub>1</sub>	F <sub>2</sub>	F <sub>3</sub>	F <sub>4</sub>	F <sub>5</sub>	F <sub>6</sub>	F <sub>7</sub>	F <sub>8</sub>	F <sub>9</sub>	F <sub>10</sub>

**Resultant sentence:** *Japan to invest \$35 billion to finance infrastructure projects*  
*and smart cities in India over the next 5 years*

**Fig. 3.** Fusion of PTE-III category sentence pair

generated summary. Average F scores of ROUGE metrics for all the summaries are shown in Table 3.

## 4.2 Experimental Setting

Sentences include Twitter specific tokens like URL, Hashtag, composite word, abbreviation and Usermention etc. In order to normalization of tokens, we used social media text processing tool<sup>3</sup> developed based on the work [1] for word segmentation (for splitting hashtags) and spell correction. All the processed sentences for an event is given to pointer-generator summarization

<sup>3</sup><https://github.com/cbaziotis/ekphrasis>

tool<sup>4</sup> for summary generation. We have used a pre-trained model<sup>5</sup> having 256 dimensional hidden states, 128 dimensional word embedding, vocabulary of 50k word size and 223000 training iterations in this work for summarization tool. We set input article token limit to 600 tokens and summary token limit to 100 tokens.

## 4.3 Evaluation Metrics and Result

To evaluate the performance of our proposed approach, we used ROUGE metrics which represent the quality of machine generated

<sup>4</sup><https://github.com/abisee/pointer-generator>

<sup>5</sup><https://github.com/abisee/pointer-generator>

**Table 3.** IAA scores (ROUGE) of human generated summaries

Event	Average F-Score			
	ROUGE-1	ROUGE-2	ROUGE-L	ROUGE-SU4
25	0.63	0.56	0.55	0.57

summaries comparing with human generated summaries [19]. ROUGE evaluation tool<sup>6</sup> developed based on the work [8] is used to measure ROUGE-1, ROUGE-2, ROUGE-L and ROUGE-SU4 F scores. During comparison, we considered stop words to reduce the impact of high overlap.

We conduct two fold experiments to evaluate our approach. In one fold, we report the performance of our approach (A1) in Table 4, comparing with human generated summaries. We also compare our approach with another abstract summarization tool (A2)<sup>7</sup> implemented based on sequence to sequence model with attention as proposed in [24]. Detail comparative result is reported in Table 4. The result shows that our approach outperforms over compared approach significantly.

In another fold, we compared our approach generated summaries (A1) with the summaries (A3) generated from the top ranked 20 tweets without SBD and filtering out redundant information, compared with summaries (A4) generated from the top ranked 20 tweets only without any pre-processing or normalization. The performance of comparisons are reported in Table 5. Result shows that SBD phase and PTE phase improves the quality of summaries.

Performance analysis of all the experimented approaches are also shown in figure 4. From the figure it is observed that, abstract summarization from tweets without any kind of pre-processing returns poor quality summary. This may be due to the nature of social media text and presence of Twitter specific tokens. Detailed analysis of the result is presented in next section.

<sup>6</sup><https://github.com/kavgan/ROUGE-2.0>

<sup>7</sup><https://github.com/zwc12/Summarization>

## 5 Discussion

After analyzing results, we observe that the score of ROUGE-1 is maximum and ROUGE-2 is least among all the evaluation metrics for our proposed approach as well as other experimented approaches. This may be due to the choice of word combination or word merging in the system and human generated summaries. Bi-gram overlap score sometimes decreases due to inappropriate bi-gram or order of words in sentence. ROUGE-SU4 score is also close to ROUGE-2 due to the similar kind of reason where ROUGE-SU4 measure is based on Skip-bigram and uni-gram based co-occurrence statistics.

We also observe that summaries generated from top ranked tweets without SBD and PTE include incomplete sentence or sentence with improper punctuation rather complete readable sentence. In some cases, partial redundant information is also present in the summaries. For example,

**Summary using SBD and PTE:** *modi means master of developing india modi. 100 days are you satisfied with the performance so far of modi govt . the downfall of the movement ( of govt ) remains , at best , unclear .*

**Summary without SBD and PTE:** *modi 100 days .. modi government . modi will be successful pm & lead the nation modi. modi 100 days the direction of the movement ( of govt ) .*

After through manual verification of newly generated summaries, we observe following errors:

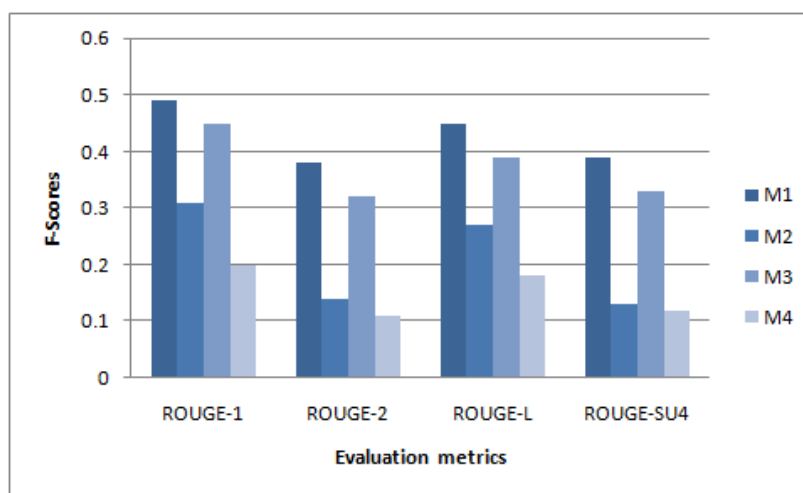
1. Our approach generated abstract summaries from extracted most relevant and informative

**Table 4.** Comparative result of proposed approach

Approach	ROUGE-1	ROUGE-2	ROUGE-L	ROUGE-SU4
A1	<b>0.49</b>	0.38	0.45	0.39
A2	0.31	0.14	0.27	0.13

**Table 5.** Performance of proposed approach on different set of data

Approach	ROUGE-1	ROUGE-2	ROUGE-L	ROUGE-SU4
A1	<b>0.49</b>	0.38	0.45	0.39
A3	0.45	0.32	0.39	0.33
A4	0.20	0.11	0.18	0.12

**Fig. 4.** Performance analysis of compared approaches

sentences. Thus, summary sentences are mostly close to the source sentence style. However, human-written summaries are mostly composed by understanding the original sentences using different vocabulary and structure. For example:

**Human generated:** there was an earthquake in northern california of magnitude 6.0 .

**System generated:** i learned that there was one earthquake in northern california of magnitude 6.0 .

with such texts remains a challenge for our approach. For example:

**Source sentence 01:** usa + europe + asia must help africa , liberia cannot cope alone with ebola outbreak

**Source sentence 02:** this is the 8 y / o girl killed by israel in gaza this morning .

## 6 Conclusion

Social media like Twitter is a great source of information for any happening events nowadays. Recent trend of information dissemination in Twitter makes this platform more and more popular than any news media. The main reason behind that is

2. Twitter text often contain grammar, self created acronym and dictation errors. Dealing

fast and diverse information sharing. In this paper, we proposed an approach to summarize any event from Twitter by extracting sentences and exploring partially entailed information in sentences to avoid maximum amount redundant information. Through experimental result we showed that our proposed approach can achieve comparable result in line with the earlier research work on Twitter event summarization. Our abstractive summary cover more information within limited words and able to give a quick overview of the event.

In recent days, social media platforms are also suffering from fake information as well as information overloading problem for any happening event. To validate a tweet's trustability is a big research issue. Summary of any event must not include any information about the event which is fake or least trustable. This type of summary will create unnecessary misconception about an event. In light of the proposed approach, future scope of the work is to measure trustworthiness of tweet content to be included in the summary.

## References

1. Baziotis, C., Pelekis, N., & Doukeridis, C. (2017). DataStories at SemEval-2017 task 4: Deep LSTM with attention for message-level and topic-based sentiment analysis. In *Proceedings of the 11th International Workshop on Semantic Evaluation (SemEval-2017)*. Association for Computational Linguistics, Vancouver, Canada, 747–754.
2. Belkaroui, R. & Faiz, R. (2017). Conversational based method for tweet contextualization. *Vietnam Journal of Computer Science*, 1–10. ISSN 2196-8896. doi:10.1007/s40595-016-0092-y.
3. Castillo, C., Mendoza, M., & Poblete, B. (2011). Information credibility on twitter. In *Proceedings of the 20th international conference on World wide web*. ACM, 675–684.
4. Chakrabarti, D. & Punera, K. (2011). Event summarization using tweets. In *Proceedings of the Fifth International Conference on Weblogs and Social Media, Barcelona, Catalonia, Spain, July 17-21, 2011*. 66–73.
5. Chellal, A., Boughanem, M., & Dousset, B. (2016). Multi-criterion real time tweet summarization based upon adaptive threshold. In *2016 IEEE/WIC/ACM International Conference on Web Intelligence (WI)*. 264–271. doi:10.1109/WI.2016.0045.
6. Das, A. & Rudrapal, D. (2014). Exploring the partial textual entailment problem for bengali news texts. *Research in Computing Science*, 86, 43–52.
7. Duan, Y., Chen, Z., Wei, F., Zhou, M., & Shum, H. (2012). Twitter topic summarization by ranking tweets using social influence and content quality. In *COLING 2012, 24th International Conference on Computational Linguistics, Proceedings of the Conference: Technical Papers, 8-15 December 2012, Mumbai, India*. 763–780.
8. Ganesan, K. (2018). Rouge 2.0: Updated and improved measures for evaluation of summarization tasks. *arXiv preprint arXiv:1803.01937*.
9. Gao, D., Li, W., & Zhang, R. (2013). Sequential summarization: A new application for timely updated twitter trending topics. In *ACL (2). The Association for Computer Linguistics*. ISBN 978-1-937284-51-0, 567–571.
10. Gimpel, K., Schneider, N., O'Connor, B., Das, D., Mills, D., Eisenstein, J., Heilman, M., Yogatama, D., Flanigan, J., & Smith, N. A. (2011). Part-of-speech tagging for twitter: Annotation, features, and experiments. In *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies: Short Papers - Volume 2*, HLT '11. Association for Computational Linguistics, Stroudsburg, PA, USA. ISBN 978-1-932432-88-6, 42–47.
11. Gupta, A., Kumaraguru, P., Castillo, C., & Meier, P. (2014). Tweetcred: Real-time credibility assessment of content on twitter. In *International Conference on Social Informatics*. Springer, 228–243.
12. Gupta, M., Zhao, P., & Han, J. (2012). Evaluating event credibility on twitter. In *Proceedings of the 2012 SIAM International Conference on Data Mining*. SIAM, 153–164.
13. Harabagiu, S. M. & Hickl, A. (2011). Relevance modeling for microblog summarization. In *Proceedings of the Fifth International Conference on Weblogs and Social Media, Barcelona, Catalonia, Spain, July 17-21, 2011*. 514–517.
14. He, R. & Duan, X. (2018). Twitter summarization based on social network and sparse reconstruction. In *The Thirty-Second AAAI Conference on Artificial Intelligence*. 5787–5794.

15. Huang, Y., Shen, C., & Li, T. (2018). Event summarization for sports games using twitter streams. *World Wide Web*, 21(3), 609–627.
16. Inouye, D. (2010). Multiple post microblog summarization. In *Research Final Rep.* 34–40.
17. Joachims, T. (2006). Training linear svms in linear time. In *Proceedings of the 12th ACM SIGKDD international conference on Knowledge discovery and data mining*. ACM, 217–226.
18. Judd, J. & Kalita, J. (2013). Better twitter summaries? In *Human Language Technologies: Conference of the North American Chapter of the Association of Computational Linguistics, Proceedings, June 9-14, 2013, Westin Peachtree Plaza Hotel, Atlanta, Georgia, USA*. 445–449.
19. Lin, C.-Y. (2004). Rouge: A package for automatic evaluation of summaries. *Text Summarization Branches Out*.
20. Mallela, D., Ahlers, D., & Pera, M. S. (2017). Mining twitter features for event summarization and rating. In *Proceedings of the International Conference on Web Intelligence*. ACM, 615–622.
21. Marcus, A., Bernstein, M. S., Badar, O., Karger, D. R., Madden, S., & Miller, R. C. (2011). Twitinfo: aggregating and visualizing microblogs for event exploration. In *Proceedings of the SIGCHI conference on Human factors in computing systems*. ACM, 227–236.
22. Mihalcea, R. & Tarau, P. (2004). TextRank: Bringing order into text. In *Proceedings of the 2004 Conference on Empirical Methods in Natural Language Processing*. Association for Computational Linguistics, Barcelona, Spain, 404–411.
23. Miranda-Jimenez, S., Gelbukh, A., & Sidorov, G. (2013). Summarizing conceptual graphs for automatic summarization task. In *Conceptual Structures for STEM Research and Education*. Springer Berlin Heidelberg, Berlin, Heidelberg, 245–253. doi:10.1007/978-3-642-35786-2\_18.
24. Nallapati, R., Xiang, B., & Zhou, B. (2016). Sequence-to-sequence rnns for text summarization. *CoRR*, abs/1602.06023.
25. Nasser Alsaedi, O. R., Pete Burnap (2016). Automatic summarization of real world events using twitter. In *The Tenth International AAAI Conference on Web and Social Media, Cologne, Germany, from 17-20 May 2016*. 511–514.
26. Olariu, A. (2014). Efficient online summarization of microblogging streams. In *Proceedings of the 14th Conference of the European Chapter of the Association for Computational Linguistics, EACL 2014, April 26-30, 2014, Gothenburg, Sweden*. 236–240.
27. Pennington, J., Socher, R., & Manning, C. D. (2014). Glove: Global vectors for word representation. In *Empirical Methods in Natural Language Processing (EMNLP)*. 1532–1543.
28. Pirró, G. & Euzenat, J. (2010). A feature and information theoretic framework for semantic similarity and relatedness. In *9th International Semantic Web Conference (ISWC2010)*. Springer, 615–630.
29. Qu, Q., Liu, S., Zhu, F., & Jensen, C. S. (2016). Efficient online summarization of large-scale dynamic networks. *IEEE Transactions on Knowledge and Data Engineering*, 28(12), 3231–3245. ISSN 1041-4347. doi:10.1109/TKDE.2016.2601611.
30. Ritter, A., Clark, S., Mausam, & Etzioni, O. (2011). Named entity recognition in tweets: An experimental study. In *EMNLP, EMNLP '11*. Association for Computational Linguistics. ISBN 978-1-937284-11-4, 1524–1534.
31. Ritter, A., Mausam, Etzioni, O., & Clark, S. (2012). Open domain event extraction from twitter. In *Proceedings of the 18th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, KDD '12*. ACM, New York, NY, USA. ISBN 978-1-4503-1462-6, 1104–1112. doi:10.1145/2339530.2339704.
32. Rudra, K., Banerjee, S., Ganguly, N., Goyal, P., Imran, M., & Mitra, P. (2016). Summarizing situational tweets in crisis scenario. In *Proceedings of the 27th ACM Conference on Hypertext and Social Media, HT '16*. ACM, New York, NY, USA. ISBN 978-1-4503-4247-6, 137–147. doi:10.1145/2914586.2914600.
33. Rudrapal, D. & Das, A. (2017). Measuring the limit of semantic divergence for english tweets. In *Proceedings of the International Conference Recent Advances in Natural Language Processing, RANLP 2017*. INCOMA Ltd., 618–624. doi:10.26615/978-954-452-049-6\_080.
34. Rudrapal, D., Das, A., & Bhattacharya, B. (2018). A survey on automatic twitter event summarization. *JIPS*, 14, 79–100.

35. Rudrapal, D., Das, A., & Bhattacharya, B. (2019). Recognition of partial textual entailment for Indian social media text. *Computación y Sistemas*, 23(1).
36. Rudrapal, D., Jamatia, A., Chakma, K., Das, A., & Gambäck, B. (2015). Sentence boundary detection for social media text. In *Proceedings of the 12th International Conference on Natural Language Processing*. 254–260.
37. Searle, J. R. (1975). Indirect speech acts. In Cole, P. & Morgan, J., editors, *Syntax and Semantics 3: Speech Acts*. Academic Press, New York, 59–82.
38. See, A., Liu, P. J., & Manning, C. D. (2017). Get to the point: Summarization with pointer-generator networks. *arXiv preprint arXiv:1704.04368*.
39. Shapira, O., Ronen, H., Adler, M., Amsterdamer, Y., Bar-Ilan, J., & Dagan, I. (2017). Interactive abstractive summarization for event news tweets. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing: System Demonstrations*. 109–114.
40. Sharifi, B., anthony Hutton, M., & Kalita, J. (2010). Automatic summarization of twitter topics. In *National Workshop on Design and Analysis of Algorithm*.
41. Sharifi, B., Hutton, M.-A., & Kalita, J. (2010). Summarizing microblogs automatically. In *Human Language Technologies: The 2010 Annual Conference of the North American Chapter of the Association for Computational Linguistics*, HLT '10. Association for Computational Linguistics, Stroudsburg, PA, USA. ISBN 1-932432-65-5, 685–688.
42. Shen, C., Liu, F., Weng, F., & Li, T. (2013). A participant-based approach for event summarization using twitter streams. In *Human Language Technologies: Conference of the North American Chapter of the Association of Computational Linguistics, Proceedings, June 9-14, 2013, Westin Peachtree Plaza Hotel, Atlanta, Georgia, USA*. 1152–1162.
43. Sidorov, G., Gelbukh, A., Gómez-Adorno, H., & Pinto, D. (2014). Soft similarity and soft cosine measure: Similarity of features in vector space model. *Computación y Sistemas*, 18(3), 491–504.
44. Wang, Z., Shou, L., Chen, K., Chen, G., & Mehrotra, S. (2015). On summarization and timeline generation for evolutionary tweet streams. *IEEE Transactions on Knowledge and Data Engineering*, 27(5), 1301–1315. ISSN 1041-4347. doi:10.1109/TKDE.2014.2345379.
45. Weng, J. & Lee, B.-S. (2011). Event detection in twitter. *ICWSM*, 11, 401–408.
46. Wu, Y., Zhang, H., Xu, B., Hao, H., & Liu, C. (2015). Tr-lda: A cascaded key-bigram extractor for microblog summarization. *International Journal of Machine Learning and Computing*, 5(3), 172–178.
47. Xu, W., Ritter, A., Callison-Burch, C., Dolan, W. B., & Ji, Y. (2014). Extracting lexically divergent paraphrases from twitter. *Transactions of the Association for Computational Linguistics*, 2, 435–448.
48. Yang, F., Liu, Y., Yu, X., & Yang, M. (2012). Automatic detection of rumor on sina weibo. In *Proceedings of the ACM SIGKDD Workshop on Mining Data Semantics*. ACM, 13.
49. Yang, X., Ghoting, A., Ruan, Y., & Parthasarathy, S. (2012). A framework for summarizing and analyzing twitter feeds. In *Proceedings of the 18th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, KDD '12. ACM, New York, NY, USA. ISBN 978-1-4503-1462-6, 370–378. doi:10.1145/2339530.2339591.
50. Zhang, R., Li, W., Gao, D., & You, O. (2013). Automatic twitter topic summarization with speech acts. *IEEE Trans. Audio, Speech & Language Processing*, 21(3), 649–658. doi:10.1109/TASL.2012.2229984.
51. Zhao, W. X., Wen, J., & Li, X. (2016). Generating timeline summaries with social media attention. *Frontiers of Computer Science*, 10(4), 702–716. doi:10.1007/s11704-015-5145-3.
52. Zhou, Y., Kanhabua, N., & Cristea, A. I. (2016). Real-time timeline summarisation for high-impact events in twitter. In *ECAI 2016 - 22nd European Conference on Artificial Intelligence, 29 August-2 September 2016, The Hague, The Netherlands - Including Prestigious Applications of Artificial Intelligence (PAIS 2016)*. 1158–1166. doi:10.3233/978-1-61499-672-9-1158.
53. Zubiaga, A., Spina, D., Amigó, E., & Gonzalo, J. (2012). Towards real-time summarization of scheduled events from twitter streams. In *Proceedings of the 23rd ACM Conference on Hypertext and Social Media*, HT '12. ACM, New York, NY, USA. ISBN 978-1-4503-1335-3, 319–320. doi:10.1145/2309996.2310053.

Article received on 05/02/2019; accepted on 04/03/2019.  
Corresponding author is Dwijen Rudrapal.



# Recognizing Musical Entities in User-generated Content

Lorenzo Porcaro<sup>1</sup>, Horacio Saggion<sup>2</sup>

<sup>1</sup> Universitat Pompeu Fabra,  
Music Technology Group, Barcelona,  
Spain

<sup>2</sup> Universitat Pompeu Fabra,  
TALN Natural Language Processing Group, Barcelona,  
Spain

{lorenzo.porcaro, horacio.saggion}@upf.edu

**Abstract.** Recognizing Musical Entities is important for Music Information Retrieval (MIR) since it can improve the performance of several tasks such as music recommendation, genre classification or artist similarity. However, most entity recognition systems in the music domain have concentrated on formal texts (e.g. artists' biographies, encyclopedic articles, etc.), ignoring rich and noisy user-generated content. In this work, we present a novel method to recognize musical entities in Twitter content generated by users following a classical music radio channel. Our approach takes advantage of both formal radio schedule and users' tweets to improve entity recognition. We instantiate several machine learning algorithms to perform entity recognition combining task-specific and corpus-based features. We also show how to improve recognition results by jointly considering formal and user-generated content.

**Keywords.** Named entity recognition, music information retrieval, user-generated content.

## 1 Introduction

The increasing use of social media and microblogging services has broken new ground in the field of Information Extraction (IE) from user-generated content (UGC). Understanding the information contained in users' content has become one of the main goals for many applications, due to the uniqueness and the variety of this data [4]. However, the highly informal and noisy status of these sources makes it difficult to apply techniques

proposed by the NLP community for dealing with formal and structured content [21].

In this work, we analyze a set of tweets related to a specific classical music radio channel, *BBC Radio 3*<sup>1</sup>, interested in detecting two types of musical named entities, *Contributor* (person related to a musical work) and *Musical Work* (musical composition or recording).

The method proposed makes use of the information extracted from the radio schedule for creating links between users' tweets and tracks broadcasted. Thanks to this linking, we aim to detect when users refer to entities included into the schedule. Apart from that, we consider a series of linguistic features, partly taken from the NLP literature and partly specifically designed for this task, for building statistical models able to recognize the musical entities. To that aim, we perform several experiments with a supervised learning model, Support Vector Machine (SVM), and a recurrent neural network architecture, a bidirectional LSTM with a CRF layer (biLSTM-CRF).

The contributions of this work are summarized as follows:

- A method to recognize musical entities from user-generated content which combines contextual information (i.e. radio schedule)

<sup>1</sup>[www.twitter.com/BBCRadio3](http://www.twitter.com/BBCRadio3)

with Machine Learning models for improving the accuracy while recognizing the entities.

- The release of language resources such as an user-generated and bot-generated Twitter corpora manually annotated, usable for both MIR and NLP researches, and domain specific word embeddings.

The paper is structured as follows. In Section 2, we present a review of the previous works related to Named Entity Recognition, focusing on its application on UGC and MIR. Afterwards, in Section 3 it is presented the methodology of this work, describing the dataset and the method proposed. In Section 4, the results obtained are shown. Finally, in Section 5 conclusions are discussed.

## 2 Related Work

Named Entity Recognition (NER), or alternatively Named Entity Recognition and Classification (NERC), is the task of detecting entities in an input text and to assign them to a specific class. It starts to be defined in the early '80, and over the years several approaches have been proposed [11]. Early systems were based on handcrafted rule-based algorithms, while afterward thanks to advancements in Machine Learning techniques, probabilistic models started to be integrated into NER systems.

In particular, new developments in neural architectures have become an important resource for this task. Their main advantages are that they do not need language-specific knowledge resources [6], and they are robust to the noisy and short nature of social media messages [7]. Indeed, according to a performance analysis of several Named Entity Recognition and Linking systems presented in [1], it has been found that poor capitalization is one of the main issues when dealing with microblog content. Apart from that, typographic errors and the ubiquitous occurrence of out-of-vocabulary (OOV) words also cause drops in NER recall and precision, together with shortenings and slang, particularly pronounced in tweets.

**Table 1.** Examples of user-generated tweets

1	No Schoenberg or Webern?? Beethoven is there but not his pno sonata op. 101??
2	Heard some of Opera 'Oberon' today... Weber... Only a little....
3	Cavalleria Rusticana...hm..from a Competition that very nearly didn't get entered!

Music Information Retrieval (MIR) is an interdisciplinary field which borrows tools of several disciplines, such as signal processing, musicology, machine learning, psychology and many others, for extracting knowledge from musical objects (be them audio, texts, etc.) [10]. In the last decade, several MIR tasks have benefited from NLP, such as sound and music recommendation [15], automatic summary of song review [23], artist similarity [22] and genre classification [12].

In the field of IE, a first approach for detecting musical named entities from raw text, based on Hidden Markov Models, has been proposed in [26]. In [13], the authors combine state-of-the-art Entity Linking (EL) systems to tackle the problem of detecting musical entities from raw texts. The method proposed relies on the *argumentum ad populum* intuition, so if two or more different EL systems perform the same prediction in linking a named entity mention, the more likely this prediction is to be correct. In detail, the off-the-shelf systems used are: DBpedia Spotlight [8], TagMe [2], Babelify [9]. Moreover, a first Musical Entity Linking, MEL<sup>2</sup> has been presented in [14] which combines different state-of-the-art NLP libraries and SimpleBrainz, an RDF knowledge base created from MusicBrainz<sup>3</sup> after a simplification process.

Furthermore, *Twitter* has also been at the center of many studies done by the MIR community. As example, for building a music recommender system [24] analyzes tweets containing keywords like *nowplaying* or *listeningto*. In [22], a similar dataset it is used for discovering cultural listening patterns.

<sup>2</sup><http://mel.mtg.upf.edu>

<sup>3</sup><https://musicbrainz.org>

Publicly available *Twitter* corpora built for MIR investigations have been created, among others the *Million Musical Tweets* dataset<sup>4</sup> [5] and the *#nowplaying* dataset<sup>5</sup> [25].

### 3 Methodology

We propose a hybrid method which recognizes musical entities in UGC using both contextual and linguistic information. We focus on detecting two types of entities:

- Contributor: person who is related to a musical work (composer, performer, conductor, etc).
- Musical Work: musical composition or recording (symphony, concerto, overture, etc).

As case study, we have chosen to analyze tweets extracted from the channel of a classical music radio, *BBC Radio 3*. The choice to focus on classical music has been mostly motivated by the particular discrepancy between the informal language used in the social platform and the formal nomenclature of contributors and musical works. Indeed, users when referring to a musician or to a classical piece in a tweet, rarely use the full name of the person or of the work, as shown in Table 2.

We extract information from the radio schedule for recreating the musical context to analyze user-generated tweets, detecting when they are referring to a specific work or contributor recently played. We manage to associate to every track broadcasted a list of entities, thanks to the tweets automatically posted by the *BBC Radio3 Music Bot*<sup>6</sup>, where it is described the track actually on air in the radio. In Table 3, examples of bot-generated tweets are shown.

Afterwards, we detect the entities on the user-generated content by means of two methods: on one side, we use the entities extracted from the radio schedule for generating candidates entities in the user-generated tweets, thanks to a matching algorithm based on time proximity and string similarity. On the other side, we create a statistical model capable of detecting entities directly from

**Table 2.** Example of entities annotated (*top*) and corresponding formal forms (*down*), from the user-generated tweet (1) in Table 1

<i>Informal form</i>	
	Schoenberg
	Webern
	Beethoven
	pno sonata op. 101
<i>Formal form</i>	
	Arnold Franz Walter Schoenberg
	Anton Friedrich Wilhelm Webern
	Ludwig Van Beethoven
	Piano Sonata No. 28 in A major, Op. 101

**Table 3.** Examples of bot-generated tweets

1	Now Playing Joaquín Rodrigo, Goran Listes - 3 Piezas españolas for guitar #joaquinrodrigo, #goranlistes
2	Now Playing Robert Schumann, Luka Mitev - Phantasiestücke, Op 73 #robertschumann, #lukamitev
3	Now Playing Pyotr Ilyich Tchaikovsky, MusicAeterna - Symphony No.6 in B minor #pyotrilyichtchaikovsky, #musicaeterna

the UGC, aimed to model the informal language of the raw texts. In Figure 1, an overview of the system proposed is presented.

#### 3.1 Dataset

In May 2018, we crawled Twitter using the Python library *Tweepy*<sup>7</sup>, creating two datasets on which *Contributor* and *Musical Work* entities have been manually annotated, using Inside-outside-beginning tags [19].

The first set contains user-generated tweets related to the *BBC Radio 3* channel. It represents the source of user-generated content on which we aim to predict the named entities. We create it filtering the messages containing hashtags related to *BBC Radio 3*, such as *#BBCRadio3* or *#BBCR3*. We obtain a set of 2,225 unique user-generated tweets.

<sup>4</sup><http://www.cp.jku.at/datasets/MMTD>

<sup>5</sup><https://zenodo.org/record/2594483>

<sup>6</sup><https://twitter.com/BBCR3MusicBot>

<sup>7</sup><http://www.tweepy.org>

**Table 4.** Example of musical named entities annotated

Beethoven	is	there	but
B-CONTR	O	O	O
not	his	pno	sonata
O	O	B-WORK	I-WORK

The second set consists of the messages automatically generated by the *BBC Radio 3 Music Bot*. This set contains 5,093 automatically generated tweets, thanks to which we have recreated the schedule.

In Table 5, the amount of tokens and entities annotated are reported for the two datasets. For evaluation purposes, both sets are split in a training part (80%) and two test sets (10% each one) randomly chosen. Within the user-generated corpora, entities annotated are only about 5% of the whole amount of tokens. In the case of the automatically generated tweets, the percentage is significantly greater and entities represent about the 50%.

### 3.2 NER System

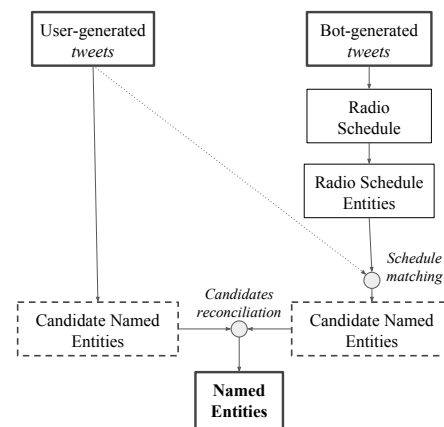
According to the literature reviewed, state-of-the-art NER systems proposed by the NLP community are not tailored to detect musical entities in user-generated content. Consequently, our first objective has been to understand how to adapt existing systems for achieving significant results in this task.

In the following sections, we describe separately the features, the word embeddings and the models considered. All the resources used are publicly available<sup>8</sup>.

#### 3.2.1 Features' Description

We define a set of features for characterizing the text at the token level. We mix standard linguistic features, such as Part-Of-Speech (POS) and chunk tag, together with several gazetteers specifically built for classical music, and a series of features representing tokens' left and right context.

<sup>8</sup><https://github.com/MTG/music-ner>

**Fig. 1.** Overview of the NER system proposed

For extracting the POS and the chunk tag we use the Python library *twitter\_nlp*<sup>9</sup>, presented in [21].

In total, we define 26 features for describing each token: 1) POS tag; 2) Chunk tag; 3) Position of the token within the text, normalized between 0 and 1; 4) If the token starts with a capital letter; 5) If the token is a digit. Gazetteers: 6) Contributor first names; 7) Contributor last names; 8) Contributor types ("soprano", "violinist", etc.); 9) Classical work types ("symphony", "overture", etc.); 10) Musical instruments; 11) Opus forms ("op", "opus"); 12) Work number forms ("no", "number"); 13) Work keys ("C", "D", "E", "F", "G", "A", "B", "flat", "sharp"); 14) Work Modes ("major", "minor", "m"). Finally, we complete the tokens' description including as token's features the surface form, the POS and the chunk tag of the previous and the following two tokens (12 features).

#### 3.2.2 Word Embedding

We consider two sets of GloVe word embeddings [16] for training the neural architecture, one pre-trained with 2B of tweets, publicly downloadable<sup>10</sup>, one trained with a corpora of 300K tweets collected during the 2014-2017 *BBC Proms Festivals* and disjoint from the data used in our experiments.

<sup>9</sup>[https://github.com/aritter/twitter\\_nlp](https://github.com/aritter/twitter_nlp)

<sup>10</sup><https://github.com/stanfordnlp/GloVe>

**Table 5.** Tokens' distributions within the two datasets: user-generated tweets (*top*) and bot-generated tweets (*bottom*)

	Training	TestA	TestB
<i>Contributor</i>	1.069 (3,12%)	119 (2,96%)	127 (2,97%)
<i>Musical Work</i>	964 (2,81%)	118 (2,93%)	163 (3,81%)
Total tokens	34.247	4.016	4.275
<hr/>			
<i>Contributor</i>	15.162 (27,50%)	1.852 (22,93%)	1.879 (27,30%)
<i>Musical Work</i>	12.904 (23,40%)	1.625 (23,56%)	1.689 (24,48%)
Total tokens	55.122	6.897	6.881

### 3.2.3 Models

The first model considered for this task has been the John Platt's sequential minimal optimization algorithm for training a support vector classifier [17], implemented in WEKA [3]. Indeed, in [18] results shown that SVM outperforms other machine learning models, such as Decision Trees and Naive Bayes, obtaining the best accuracy when detecting named entities from the user-generated tweets.

However, recent advances in Deep Learning techniques have shown that the NER task can benefit from the use of neural architectures, such as biLSTM-networks [6, 7]. We use the implementation<sup>11</sup> proposed in [20] for conducting three different experiments. In the first, we train the model using only the word embeddings as feature. In the second, together with the word embeddings we use the POS and chunk tag. In the third, all the features previously defined are included, in addition to the word embeddings. For every experiment, we use both the pre-trained embeddings and the ones that we created with our *Twitter* corpora. In section 4, results obtained from the several experiments are reported.

### 3.3 Schedule Matching

The bot-generated tweets present a predefined structure and a formal language, which facilitates the entities detection. In this dataset, our goal is to assign to each track played on the radio, represented by a tweet, a list of entities extracted from the tweet raw text. For achieving that, we experiment with the algorithms and features

presented previously, obtaining an high level of accuracy, as presented in section 4. The hypothesis considered is that when a radio listener posts a tweet, it is possible that she is referring to a track which has been played a relatively short time before. In this cases, we want to show that knowing the radio schedule can help improving the results when detecting entities.

Once assigned a list of entities to each track, we perform two types of matching. Firstly, within the tracks we identify the ones which have been played in a fixed range of time ( $t$ ) before and after the generation of the user's tweet. Using the resulting tracks, we create a list of candidates entities on which performing string similarity. The score of the matching based on string similarity is computed as the ratio of the number of tokens in common between an entity and the input tweet, and the total number of token of the entity.

In order to exclude trivial matches, tokens within a list of stop words are not considered while performing string matching. The final score is a weighted combination of the string matching score and the time proximity of the track, aimed to enhance matches from tracks played closer to the time when the user is posting the tweet.

The performance of the algorithm depends, apart from the time proximity threshold  $t$ , also on other two thresholds related to the string matching, one for the *Musical Work* ( $w$ ) and one for the *Contributor* ( $c$ ) entities. It has been necessary for avoiding to include candidate entities matched against the schedule with a low score, often source of false positives or negatives. Consequently, as last step *Contributor* and *Musical Work* candidates entities with respectively a string matching score

<sup>11</sup><https://github.com/UKPLab/emnlp2017-bilstm-cnn-crf>

lower than  $c$  and  $w$ , are filtered out. In Figure 2, an example of *Musical Work* entity recognized in an user-generated tweet using the schedule information is presented.

### 3.3.1 Candidates Reconciliation

The entities recognized from the schedule matching are joined with the ones obtained directly from the statistical models. In the joined results, the criteria is to give priority to the entities recognized from the machine learning techniques. If they do not return any entities, the entities predicted by the schedule matching are considered. Our strategy is justified by the poorer results obtained by the NER based only on the schedule matching, compared to the other models used in the experiments, to be presented in the next section.

## 4 Results

The performances of the NER experiments are reported separately for three different parts of the system proposed.

Table 6 presents the comparison of the various methods while performing NER on the bot-generated corpora and the user-generated corpora. Results shown that, in the first case, in the training set the F1 score is always greater than 97%, with a maximum of 99.65%. With both test sets performances decrease, varying between 94-97%. In the case of UGC, comparing the F1 score we can observe how performances significantly decrease. It can be considered a natural consequence of the complex nature of the users' informal language in comparison to the structured message created by the bot.

In Table 7, results of the schedule matching are reported. We can observe how the quality of the linking performed by the algorithm is correlated to the choice of the three thresholds. Indeed, the *Precision* score increase when the time threshold decrease, admitting less candidates as entities during the matching, and when the string similarity thresholds increase, accepting only candidates with an higher degree of similarity. The behaviour of the *Recall* score is inverted.

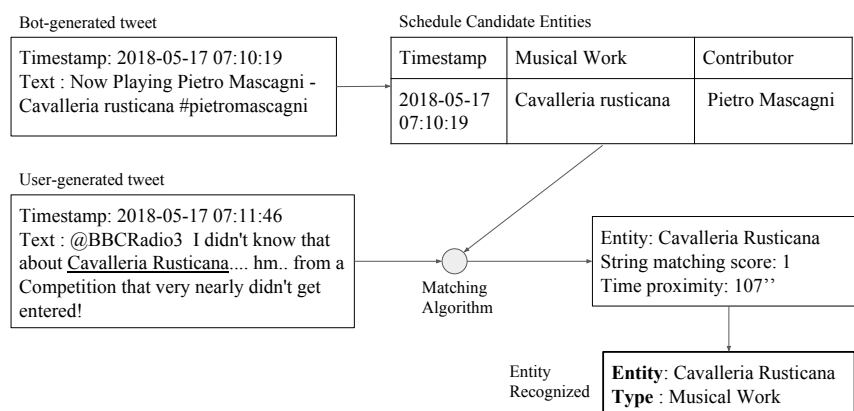
Finally, we test the impact of using the schedule matching together with a biLSTM-CRF network. In this experiment, we consider the network trained using all the features proposed, and the embeddings not pre-trained. Table 8 reports the results obtained. We can observe how generally the system benefits from the use of the schedule information. Especially in the testing part, where the neural network recognizes with less accuracy, the explicit information contained in the schedule can be exploited for identifying the entities at which users are referring while listening to the radio and posting the tweets.

## 5 Conclusion and Future Work

We have presented in this work a novel method for detecting musical entities from user-generated content, using a combination of linguistic and domain features with statistical models and extracting contextual information from a radio schedule. We analyzed tweets related to a classical music radio station, integrating its schedule to connect users' messages to tracks broadcasted. We focus on the recognition of two kinds of entities related to the music field, *Contributor* and *Musical Work*.

According to the results obtained, we have seen a pronounced difference between the system performances when dealing with the *Contributor* instead of the *Musical Work* entities. Indeed, the former type of entity has been shown to be more easily detected in comparison to the latter, and we identify several reasons behind this fact. Firstly, *Contributor* entities are less prone to be shorten or modified, but due to their length *Musical Work* entities often represent only a part of the complete title of a musical piece.

Furthermore, *Musical Work* titles are typically composed by more tokens, including common words which can be easily misclassified. The low performances obtained in the case of *Musical Work* entities can be a consequences of these observations. On the other hand, when referring to a *Contributor* users often use only the surname, but in most of the cases it is enough for the system to recognizing the entities.



**Fig. 2.** Example of the workflow for recognizing entities in UGC using the information from the radio schedule

**Table 6.** F1 score for *Contributor* (C) and *Musical Work* (MW) entities recognized from user-generated tweets (*top*) and top-generated tweets (*bottom*)

Model	Features	GloVe vectors	Training		TestA		TestB	
			C	MW	C	MW	C	MW
SVM	all	–	95.44	80.80	64.91	<b>33.48</b>	61.02	36.21
biLSTM-CRF	–	trained	79.09	51.51	60.00	26.66	67.02	31.48
		pre-trained	85.51	69.28	70.00	33.33	<b>71.26</b>	32.08
biLSTM-CRF	POS+chunk	trained	79.37	50.90	61.23	28.98	62.03	<b>40.00</b>
		pre-trained	73.51	37.28	<b>71.62</b>	25.00	63.74	25.53
biLSTM-CRF	all	trained	97.42	<b>88.92</b>	66.22	28.17	69.11	36.36
		pre-trained	<b>98.46</b>	87.35	68.79	23.68	70.41	29.51
SVM	all	–	99.12	97.70	97.74	94.32	97.88	95.42
biLSTM-CRF	–	trained	98.95	97.07	98.06	92.99	98.33	95.59
		pre-trained	99.34	94.94	97.88	91.40	98.27	92.35
biLSTM-CRF	POS+chunk	trained	<b>99.94</b>	98.28	97.99	<b>94.68</b>	98.03	<b>95.97</b>
		pre-trained	99.69	97.23	98.12	93.30	98.49	93.61
biLSTM-CRF	all	trained	99.80	98.22	97.70	91.99	98.36	94.48
		pre-trained	99.90	<b>99.40</b>	<b>98.24</b>	90.46	<b>98.78</b>	94.23

From the experiments we have seen that generally the biLSTM-CRF architecture outperforms the SVM model. The benefit of using the whole set of features is evident in the training part, but while testing the inclusion of the features not always leads to better results.

In addition, some of the features designed in our experiments are tailored to the case of classical

music, hence they might not be representative if applied to other fields. We do not exclude that our method can be adapted for detecting other kinds of entity, but it might be needed to redefine the features according to the case considered.

Similarly, it has not been found a particular advantage of using the pre-trained embeddings instead of the one trained with our corpora.

**Table 7.** Precision (P), Recall (R) and F1 score for Contributor (C) and Musical Work (MW) of the schedule matching algorithm.  $w$  indicates the Musical Work string similarity threshold,  $c$  indicates the Contributor string similarity threshold and  $t$  indicates the time proximity threshold in seconds

$w$	$c$		$t=800$			$t=1000$			$t=1200$		
			P	R	F1	P	R	F1	P	R	F1
0.33	0.33	C	72.49	16.49	26.87	69.86	17.57	28.08	68.66	<b>17.93</b>	<b>28.43</b>
		MW	26.42	4.78	8.10	26.05	5.29	8.79	23.66	5.29	8.65
0.33	0.5	C	<b>76.77</b>	14.32	24.14	74.10	15.64	25.83	73.89	16.00	26.30
		MW	27.1	4.95	8.37	26.67	5.46	<b>9.06</b>	24.24	<b>5.46</b>	8.91
0.5	0.5	C	<b>76.77</b>	14.32	24.14	74.71	15.64	25.87	73.89	16.00	26.30
		MW	<b>30.43</b>	4.78	8.26	30.30	5.12	8.76	27.52	5.12	8.63

**Table 8.** Precision (P), Recall (R) and F1 score for Contributor (C) and Musical Work (MW) entities recognized from user-generated tweets using the biLSTM-CRF network together with the schedule matching. The thresholds used for the matching are  $t=1200$ ,  $w=0.5$ ,  $c=0.5$

			Training			TestA			TestB		
			P	R	F1	P	R	F1	P	R	F1
biLSTM-CRF	C		<b>98.22</b>	96.64	<b>97.42</b>	69.01	63.64	66.22	<b>67.35</b>	70.97	<b>69.11</b>
	MW		<b>91.54</b>	86.44	<b>88.92</b>	<b>43.48</b>	20.83	28.17	<b>45.83</b>	30.14	36.36
biLSTM-CRF + Sch. Matcher	C		95.92	<b>97.81</b>	96.86	<b>74.19</b>	<b>71.88</b>	<b>73.02</b>	63.29	<b>74.63</b>	68.49
	MW		87.33	<b>87.03</b>	87.18	38.46	<b>22.73</b>	<b>28.57</b>	42.55	<b>32.26</b>	<b>36.70</b>

Furthermore, we verified the statistical significance of our experiment by using Wilcoxon Rank-Sum Test, concluding that there have been not significant difference between the various model considered while testing.

The information extracted from the schedule also presents several limitations. In fact, the hypothesis that a tweet is referring to a track broadcasted is not always verified. Even if it is common that radio listeners do comments about tracks played, or give suggestion to the radio host about what they would like to listen, it is also true that they might refer to a Contributor or Musical Work unrelated to the radio schedule.

## Acknowledgements

This work is partially supported by the European Commission under the TROMPA project (H2020 770376), and by the Spanish Ministry of Economy and Competitiveness under the Maria de Maeztu

Units of Excellence Programme (MDM-2015-0502).

## References

1. Derczynski, L., Maynard, D., Rizzo, G., Van Erp, M., Gorrell, G., Troncy, R., Petrak, J., & Bontcheva, K. (2015). Analysis of named entity recognition and linking for tweets. *Information Processing and Management*, Vol. 51, No. 2, pp. 32–49.
2. Ferragina, P. & Scaiella, U. (2012). Fast and accurate annotation of short texts with Wikipedia pages. *IEEE Software*, Vol. 29, No. 1, pp. 70–75.
3. Frank, E., Hall, M. A., & Witten, I. H. (2016). *The WEKA Workbench. Online Appendix for "Data Mining: Practical Machine Learning Tools and Techniques"*. Morgan Kaufmann, Fourth Edition.
4. Habib, M. B. & Keulen, M. V. (2014). Information Extraction for Social Media. *Workshop on Semantic Web and Information Extraction*, July, pp. 9–16.



5. Hauger, D., Schedl, M., Košir, A., & Tkalcic, M. (2013). The million musical tweets dataset: What can we learn from microblogs. *Proceedings of the 14th International Society for Music Information Retrieval Conference*, pp. 189–194.
6. Lample, G., Ballesteros, M., Subramanian, S., Kawakami, K., & Dyer, C. (2016). Neural architectures for named entity recognition. *Proceedings of the 2016 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, July, pp. 260–270.
7. Lin, B. Y., Xu, F. F., Luo, Z., & Zhu, K. Q. (2017). Multi-channel bilstm-crf model for emerging named entity recognition in social media. *Proceedings of the 3rd Workshop on Noisy User-generated Text*, pp. 160–165.
8. Mendes, P. N., Jakob, M., García-Silva, A., & Bizer, C. (2011). DBpedia Spotlight: Shedding light on the Web of documents. *Proceedings of the 7th International Conference on Semantic Systems (I-Semantics)*, Vol. 95, pp. 1–8.
9. Moro, A., Raganato, A., & Navigli, R. (2014). Entity linking meets word sense disambiguation: a unified approach. *Transactions of the Association for Computational Linguistics (TACL)*, Vol. 2, No. 0, pp. 231–244.
10. Müller, M. (2015). *Fundamentals of Music Processing: Audio, Analysis, Algorithms, Applications*. Springer Publishing Company, Incorporated, 1st edition.
11. Nadeau, D. (2007). A survey of named entity recognition and classification. *Linguisticae Investigationes. International Journal of Linguistics and Language Resources*, Vol. 30, No. 1, pp. 3–26.
12. Oramas, S., Espinosa-Anke, L., Lawlor, A., Serra, X., Saggion, H., Music Technology Group, & Universitat Pompeu Fabra (2016). Exploring customer reviews for music genre classification and evolutionary studies. *Proc. 17th International Society for Music Information Retrieval Conference*, pp. 150–156.
13. Oramas, S., Espinosa-Anke, L., Sordo, M., Saggion, H., & Serra, X. (2016). ELMD: An automatically generated entity linking gold standard dataset in the music domain. *Language Resources and Evaluation Conference - LREC*, pp. 3312–3317.
14. Oramas, S., Ferraro, A., Correya, A., & Serra, X. (2017). Mel: a music entity linking system. *Proceedings of the 18th International Society for Music Information Retrieval Conference*, Suzhou, China.
15. Oramas, S., Ostuni, V. C., Di Noia, T., Serra, X., & Di Sciascio, E. (2015). Sound and music recommendation with knowledge graphs. *ACM Transactions on Intelligent Systems and Technology*, Vol. 9, No. 4.
16. Pennington, J., Socher, R., & Manning, C. (2014). Glove: Global vectors for word representation. *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pp. 1532–1543.
17. Platt, J. (1998). Fast training of support vector machines using sequential minimal optimization. *Advances in Kernel Methods - Support Vector Learning*, MIT Press, pp. 41–65.
18. Porcaro, L. (2018). Information extraction from user-generated content in the classical music domain. *Master Thesis. Universitat Pompeu Fabra*.
19. Ramshaw, L. & Marcus, M. (1995). Text chunking using transformation-based learning. *Third Workshop on Very Large Corpora*, pp. 82–94.
20. Reimers, N. & Gurevych, I. (2017). Reporting score distributions makes a difference: Performance study of LSTM-networks for sequence tagging. *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pp. 338–348.
21. Ritter, A., Clark, S., & Etzioni, O. (2011). Named entity recognition in tweets: an experimental study. *Proceedings of the 2011 Conference on Empirical Methods in Natural Language Processing*, pp. 1524–1534.
22. Schedl, M. & Hauger, D. (2012). Mining microblogs to infer music artist similarity and cultural listening patterns. *International Conference Companion on World Wide Web (WWW)*, pp. 877.
23. Tata, S. & Di Eugenio, B. (2010). Generating fine-grained reviews of songs from album reviews. *Proceedings of the 48th Annual Meeting of the Association for Computational Linguistics*, July, pp. 1376–1385.
24. Zangerle, E., Gassler, W., & Specht, G. (2012). Exploiting twitter's collective knowledge for music recommendation. *2nd workshop on Making Sense of Microposts #MSM2012*, February, pp. 14–17.
25. Zangerle, E., Pichl, M., Gassler, W., & Specht, G. (2014). #nowplaying music dataset: Extracting listening behavior from twitter. *Proceedings of*

*the First International Workshop on Internet-Scale  
Multimedia Management - WISMM '14*, pp. 21–26.

- 26. Zhang, X., Liu, Z., Qiu, H., & Fu, Y. (2009).** A hybrid approach for chinese named entity recognition in music domain. *8th IEEE International Symposium*

*on Dependable, Autonomic and Secure Computing,  
DASC 2009*, pp. 677–681.

*Article received on 16/01/2019; accepted on 04/03/2019.  
Corresponding author is Lorenzo Porcaro.*

# Cross-Domain Failures of Fake News Detection

Maria Janicka, Maria Pszona, Aleksander Wawer

Samsung R&D Institute Poland,  
Poland

{m.janicka, m.pszona, a.wawer}@samsung.com

**Abstract.** Fake news recognition has become a prominent research topic in natural language processing. Researchers reported significant successes when applying methods based on various stylometric and lexical features and machine learning, with accuracy reaching 90%. This article is focused on answering the question: are the fake news detection models universally applicable or limited to the domain they have been trained on? We used four different, freely available English language Fake News corpora and trained models in both in-domain and cross-domain setting. We also explored and compared features important in each domain. We found that the performance in cross-domain setting degrades by 20% and sets of features important to detect fake texts differ between domains. Our conclusions support the hypothesis that high accuracy of machine learning models applied to fake news detection may be related to over-fitting, and models need to be trained and evaluated on mixed types of texts.

**Keywords.** Fake news detection, cross-domain, cross-domain failures.

## 1 Introduction

Recognizing fake news is a problem of automatically detecting misleading news stories, ones that often come from non-reputable sources. The research on fake-news detection surged since the 2016 US presidential campaign. While the most reliable approach is human fact-checking, the one we focus on in this paper is the analysis of non-lexical properties, such as psycholinguistic and stylometric variables obtained from several available tools. Non-lexical analysis is interesting because, at least in theory, it should allow to abstract from topics and domains, resulting in a more universally applicable solution.

The paper is organized as follows: Section 2 describes previous studies on the topic of fake

news detection, Section 3 describes fake news data sets, Section 4 outlines the input features and machine learning methods. Section 5 contains the results of experiments and Section 6 analysis of features.

## 2 Previous Work

As typically fake news is intentionally created to spread misinformation, their writing style slightly differ from that of reliable content. Therefore, the traditional approach to detect false content is based on linguistic features. Until now, one of the top classifiers relying on these features achieved the accuracy of up to 76% [15]. The studies revealed that the punctuation and factors related to the complexity of text - including a number of characters, words, syllables, complex words, long words and several readability metrics such as Flesch-Kincaid [6], Gunning Fog [7] and Automatic Readability Index [18], are of the highest importance. The semantic features, which can be extracted from Linguistic Inquiry and Word Count software (LIWC) [14] are also crucial. LIWC not only provides a number of words that fall into different meta-language categories such as positive emotions, analytical thinking, cognitive process but also carry out part-of-speech tagging. The successful usage of these features was demonstrated in the detection of falsified reviews [13] or prisoners lies [2].

Zheng et al. demonstrated that by using relationships between news article, its creator, subject and fake/true label, it is possible to achieve accuracy as of 0.63 [21]. They designed a diffusive network based on a set of explicit and latent features extracted exclusively from textual content.

A news article is often accompanied by visual materials like images or videos, which are rarely taken into account. Nevertheless, recent studies revealed that fake and real news exhibit different image distribution patterns. Jin et al. proposed several visual and statistical features supporting the detection of fake news [8].

Another common approach, adopted for example by B.S. Detector <sup>1</sup>, a browser extension alerting users about unreliable news source, is simply based on a curated open-source database containing an assessment of online information sources. We observed that some of the already unmasked pages that deliberately publish fake content, use redirection to different URL, so there is an overwhelming need to update this database regularly. While detecting false content, the initial step should involve the assessment of source credibility. The authors of the already mentioned database suggest 6 practical steps. For instance, checking whether the title or domain is not just a slight variation on a well-known website, verification of mentioned links, referenced sources and quotation, both aesthetic and writing style analysis.

Some studies revealed that the fake and reliable news follow different patterns of propagation in social media [5]. Interestingly, this is noticeable even at early stages of spreading, which is extremely useful in preventing the negative impact of misinformation on society [22].

### 3 Fake News Data Sets

This section describes data sets used in our experiments.

#### 3.1 Kaggle

The data set <sup>2</sup> contains news collected with B.S. Detector, a browser extension, which provides a list of unreliable sources. It is the biggest data set used in this research containing 20800 texts falling into two categories: fake and true.

<sup>1</sup><https://bsdetectector.tech/>

<sup>2</sup><https://www.kaggle.com/c/fake-news/data>

#### 3.2 LIAR

The data set [20] contains short statements manually labeled by PolitiFact (<sup>3</sup>) fact-checkers. These short texts are categorized into 6 groups, but we included only texts labeled as true together with false and pants-fire category (the most obviously fake information) labeled as fake.

#### 3.3 AMT

Fake news data was generated using crowdsourcing via Amazon Mechanical Turk. The AMT workers were asked to generate fake versions of true news collected earlier in a corpus. Each of the fake news had to mimic a journalistic style [16].

#### 3.4 Buzzfeed

This data set is created from top fake news on Facebook reported in years 2016 and 2017. The data was collected using BuzzSumo with the help of PolitiFact information and Buzzfeed own resources. Then complementary 91 real news was added.

#### 3.5 Data Set Summary

The data sets are summarized in Table 1. They differ not only in origin and length, but also in topic (although politics somehow dominates). The proportions of fake vs true are somewhat balanced.

### 4 Machine Learning

To build a comprehensive set of features we discriminated four areas of linguistic investigation. In total, we collected 279 features. The biggest subset consists of 182 General Inquirer features plus two special features from purposely designed dictionaries. The second group is built out of 61 features containing POS tags (56) and syntactic information (3). We also add 35 psycholinguistic features connected with readability and one feature containing information about text subjectivity.

In the first step, we carried out basic analysis including information about parts of speech and

<sup>3</sup><https://www.politifact.com/>

**Table 1.** Data set summary

Dataset	Size	Length	Comments
Kaggle	10 387 true 10 413 fake	medium	only politics
LIAR	2053 true 2454 mostly-true 2627 half-true 2103 barely-true 2507 false 1047 pants-fire	very short	only politics
AMT	240 true 240 fake	medium	seven domains; for each legitimate news fake news generated via Amazon Mechanical Turk (AMT)
Buzzfeed	91 true 91 fake	medium	categories: politics, breaking news, business, local news, medicine, race

syntactic structure. We used Spacy library to count the percentage of an occurrence of a given POS tag in news text. CoreNLP dependency parser was employed to measure parse tree depth together with the depth of a noun phrase.

In the next step we use General Inquirer [19] – a tool for text content analysis which provides a wide range of categories. It helps to characterize text by defining words in terms of sentiment, intensity, varying social and cognitive contexts. Categories were collected from four different sources - the Harvard IV-4 dictionary, the Lasswell value dictionary [10] - several categories were constructed based on work of Semin and Fiedler on social cognition and language [17], finally, marker categories were adapted from Kelly and Stone work on word sense disambiguation [9]. In addition to enrich existing feature set with domain-relevant terms, we created two special dictionaries containing linguistic hedges and exclusion terms. We created features from General Inquirer in fake news classifier by measuring the ratio of words in a given category to all words in a text.

Further, we enriched feature space with readability indices<sup>4</sup>. We used popular measures which represent an approximation of the level of education needed to understand a text -

<sup>4</sup><https://github.com/andreascv/readability/>

Flesh-Kincaid [6], ARI [18], Coleman-Liau [4], Gunning Fog Index [7], LIX [1], SMOG Index [11], RIX [1], Dale-Chall Index [3]. Each metric uses different premises connected with word-level and sentence-level complexity - e.g., sentence length, word length, number of syllables per word, number of long words in a text or information about part-of-speech or sentence beginnings. We also use these indicators in a stand-alone manner as a set of psycholinguistic features.

The last is a sentence-level feature – subjectivity. We used subjectivity classifier<sup>5</sup> based on bi-directional GRU to find subjective sentences in a text. Percentage of subjective sentences serves as a feature in fake news classifier.

Values of all features were normalised and four different approaches to classification task were tested. We trained support vector classifier with the linear kernel, support vector machine with stochastic gradient descent, extremely randomized decision trees and extreme gradient boosting.

## 5 Results

This section presents the results of machine learning experiments. For each data set, we treated it as a training data source, and performed

<sup>5</sup>[https://github.com/fractalego/subjectivity\\_classifier](https://github.com/fractalego/subjectivity_classifier)

a number of cross-domain experiments. We tested the obtained model on the same data set (in this case, we split data into random 80% train and 20% test subsets). We also applied it to all other data sets (in this case, we used the whole data set for training). We present each experiment in a separate table. Table 2 contains the results of models trained on Mihalcea data set [16], Table 3 illustrates the performance of models trained on Kaggle data, Table 4 shows the results of models trained on Politifact, and finally Table 5 shows the accuracy of BuzzFeed-trained models.

The results reveal that fake news detection, once models are trained and tested within the same data set, appears to be a promising problem to solve. Here, Kaggle data set is a special one: when training and testing models on it, the accuracy is astonishing (near one). The best classifiers on AMT and BuzzFeed data sets reach accuracy in the range of 0.76-0.78 difficult even for models trained on this data set, as the accuracy does not exceed 0.653.

Among classification algorithms, the one that performs best within the same data set is XGBoost. However, cross-domain application reveals that it comes at the cost of overfitting: it does not generalize well to other types of fake news data. In most cross-domain settings it is significantly outperformed by LinearSVC.

However, the most interesting observation is that in all of the cases (datasets and classifiers), applying the models to other data sets yields sharp drops of accuracy, often down to values similar to random baselines. Kaggle-trained classifiers are not better in this respect, since the accuracy ranges between 0.62 when applied to LIAR to as low as 0.4 when applied to BuzzFeed.

Surprisingly, the LinearSVC classifier trained on LIAR data set, where it reached 0.636, managed to perform better when applied to the Kaggle data (0.785).

## 6 Feature Analysis

To gain more understanding of the data, we have performed an analysis of feature distribution in each of the data sets. We have selected four

features that are both relevant and exhibit interesting patterns, and illustrated their occurrences as histograms. The features are as follows:

- Linguistic Category Model's Descriptive Action Verbs (DAV).
- Linguistic Category Model's State Verbs (SV).
- Verbs in Past Tense.
- Automated Readability Index (ARI).

Linguistic Category Model [17] is a framework for verb categorization according to their abstractness. DAV verbs correspond to the most concrete class, while SV verbs are the most abstract. According to Pennebaker et al. [12], the language of deception is linked to the higher levels of abstraction. This finding is reflected in the Kaggle data set distributions for DAV and SV verbs. True texts contain more DAVs and less SVs (are less abstract), and vice versa. This conclusion can not be observed in other three data sets.

Verbs in the past tense can also help distinguish fake and true news on the Kaggle data. Fake news less often refer to past actions and events (contain less verbs in the past tense) than true news.

Automated Readability Index (ARI) is a tool to measure language complexity and understandability. On Kaggle news, it reveals that fake news are on average less readable (a high spike in distribution) than true news. ARI is used as an example, but we can observe similar differences with respect to all of the readability measures. Those features show the most distinct values discrepancy between fake and real content, which suggests that readability plays a major role in a good in-domain performance of a classifier trained on the Kaggle data set. Again, the observation does not hold for the remaining data sets.

In the LIAR data, larger than AMT and BuzzFeed data, distribution of four features within fake news was similar to that within true news. No apparent patterns could be observed. Feature distributions in AMT and BuzzFeed data are similar, expectedly the amount of noise increases with decreasing size of the corpora.

**Table 2.** Accuracy on AMT data set

Classifier	Training set	AMT	Kaggle	LIAR	Buzzfeed
LinearSVC	AMT	0.667	0.506	0.549	0.626
SGDClassifier	AMT	0.675	0.466	0.535	0.626
ExtraTreesClassifier	AMT	0.733	0.517	0.535	0.495
XGBoost	AMT	0.767	0.42	0.463	0.588

**Table 3.** Accuracy on Kaggle data set

Classifier	Training set	Kaggle	LIAR	AMT	Buzzfeed
LinearSVC	Kaggle	0.974	0.569	0.473	0.33
SGDClassifier	Kaggle	0.973	0.586	0.465	0.308
ExtraTreesClassifier	Kaggle	0.962	0.628	0.506	0.401
XGBoost	Kaggle	0.977	0.628	0.483	0.352

**Table 4.** Accuracy on LIAR data set

Classifier	Training set	LIAR	Kaggle	AMT	Buzzfeed
LinearSVC	LIAR	0.636	0.785	0.562	0.61
SGDClassifier	LIAR	0.533	0.468	0.517	0.555
ExtraTreesClassifier	LIAR	0.629	0.464	0.552	0.516
XGBoost	LIAR	0.653	0.486	0.496	0.511

**Table 5.** Accuracy on Buzzfeed data set

Classifier	Training set	Buzzfeed	Kaggle	LIAR	AMT
LinearSVC	Buzzfeed	0.674	0.625	0.519	0.59
SGDClassifier	Buzzfeed	0.674	0.604	0.498	0.59
ExtraTreesClassifier	Buzzfeed	0.739	0.547	0.522	0.535
XGBoost	Buzzfeed	0.783	0.554	0.586	0.567

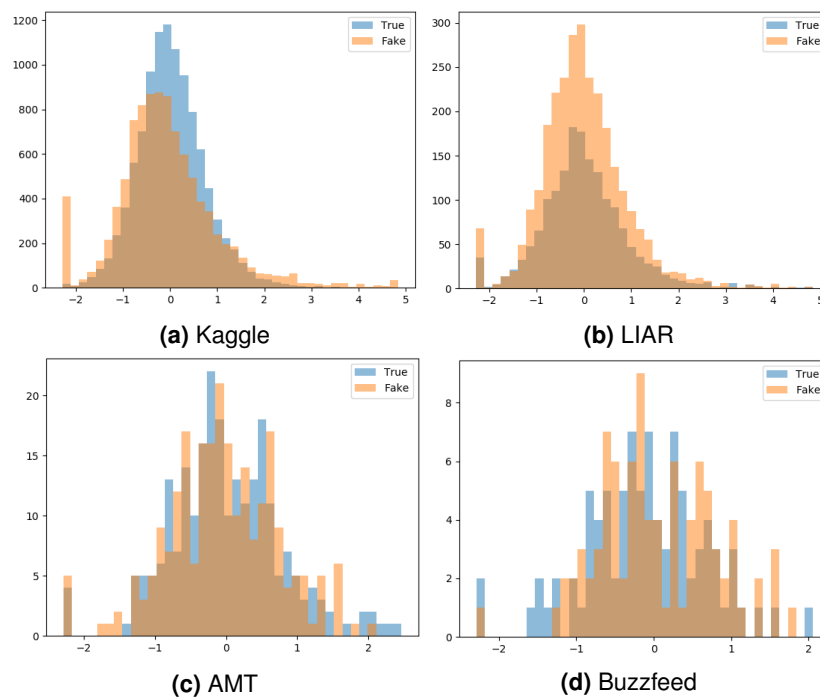
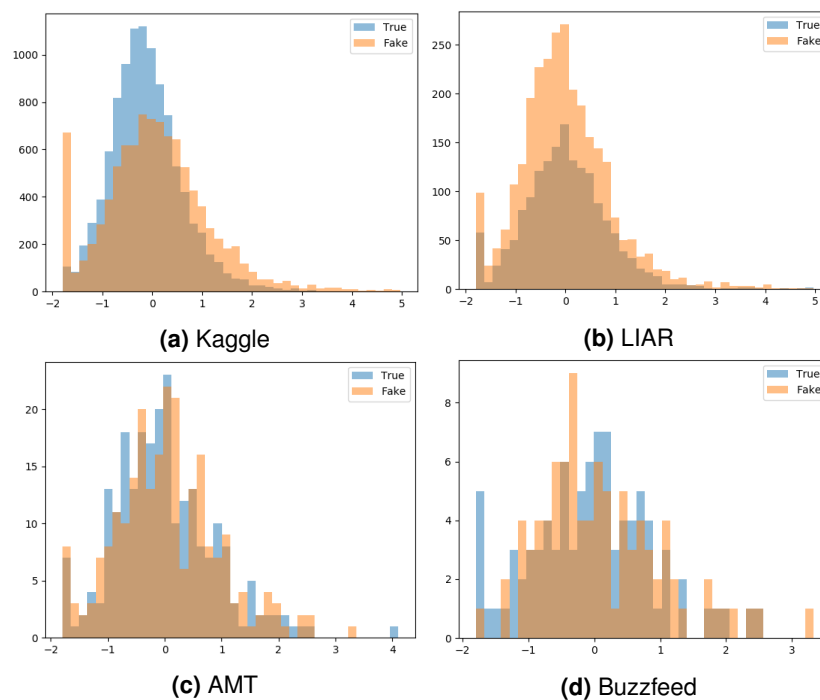
None of the three data sets seems to be usable for high performance detection of fake vs true news using stylometric and psycholinguistic features.

Contradictory, in the Kaggle data set we can see some clear differences in feature distribution, which reflects in high accuracy of classifiers trained and tested on this data set. Our hypothesis is that those news articles are of special character, as they were not collected in a manual fact-checking procedure, but were added from sources marked as unreliable.

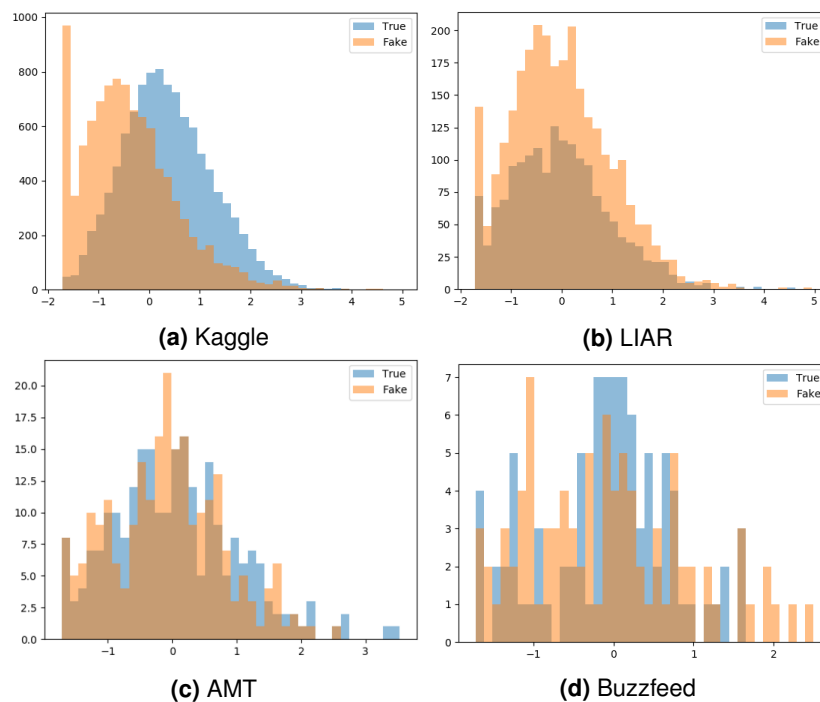
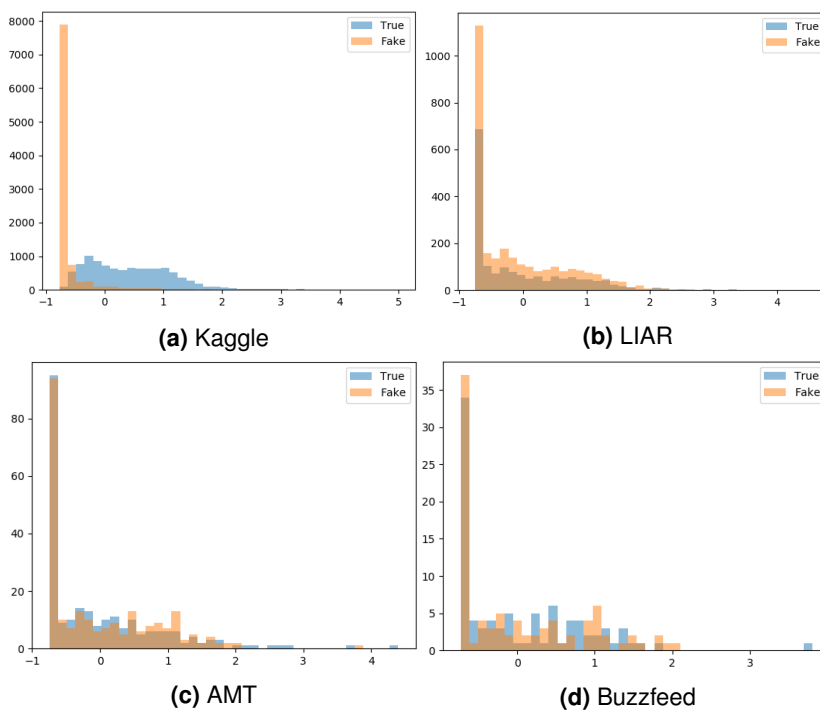
This kind of web pages are created to manipulate audience and language may differ from those of legitimate journalism.

## 7 Conclusions

Stylometric and psycholinguistic features, such as those used in our paper, were hoped to introduce universal character to fake news recognition models, and to outperform traditional machine learning based on word occurrence vectors as features. This paper argues for the opposite: successes of machine learning, measured as high accuracy in recognizing fake news texts, are strongly linked and in fact constrained to types of texts on which the models have been trained. One may think about this problem as a form of over-fitting. Also the models are hardly usable for real-life fake news detection.

**Fig. 1.** Descriptive Action Verbs (DAV)**Fig. 2.** State Verbs (SV)



**Fig. 3.** Verb - Past Tense**Fig. 4.** Automated Readability Index

Therefore, our paper outlines an important future direction for studying fake news. Instead of preparing fake news data sets which consist of texts of similar structure and the same source researchers should attempt to compile more mixed corpora, gathering short and long texts on politics, economy and many other topics, from both social media and printed sources.

The design of machine learning models should take into account postulated corpora diversity. One of the observations made in this paper was over-fitting (domain dependency) of models based on gradient boosting (eg. XGBoost) and relatively better performance of Linear SVM.

In the future, we plan to experiment with training and testing corpora compiled from varied texts with the focus on machine learning methods that prevent over-fitting. We also intend to conduct similar studies using deep learning methods.

## References

1. **Anderson, J. (1983).** Lix and rix: Variations on a little-known readability index. *Journal of Reading*, Vol. 26, No. 6, pp. 490–496.
2. **Bond, G. D. & Lee, A. Y. (2005).** Language of lies in prison: Linguistic classification of prisoners' truthful and deceptive natural language. *Applied Cognitive Psychology*, Vol. 19, No. 3, pp. 313–329.
3. **Chall, J. S. & Dale, E. (1995).** *Readability revisited: The new Dale-Chall readability formula*. Brookline Books.
4. **Coleman, M. & Liau, T. L. (1975).** A computer readability formula designed for machine scoring. *Journal of Applied Psychology*, Vol. 60, No. 2, pp. 283.
5. **Del Vicario, M., Bessi, A., Zollo, F., Petroni, F., Scala, A., Caldarelli, G., Stanley, H. E., & Quattrociocchi, W. (2016).** The spreading of misinformation online. *Proceedings of the National Academy of Sciences*, Vol. 113, No. 3, pp. 554–559.
6. **Flesch, R. (2007).** Flesch-Kincaid readability test. Retrieved October, Vol. 26, pp. 2007.
7. **Gunning, R. (1969).** The fog index after twenty years. *Journal of Business Communication*, Vol. 6, No. 2, pp. 3–13.
8. **Jin, Z., Cao, J., Zhang, Y., Zhou, J., & Tian, Q. (2017).** Novel visual and statistical image features for microblogs news verification. *IEEE transactions on multimedia*, Vol. 19, No. 3, pp. 598–608.
9. **Kelly, E. F. & Stone, P. J. (1975).** *Computer recognition of English word senses*, volume 13. North-Holland.
10. **Lasswell, H. D. & Namenwirth, J. Z. (1969).** The Lasswell value dictionary. *New Haven*.
11. **Mc Laughlin, G. H. (1969).** SMOG grading – a new readability formula. *Journal of reading*, Vol. 12, No. 8, pp. 639–646.
12. **Newman, M. L., Pennebaker, J. W., Berry, D. S., & Richards, J. M. (2003).** Lying words: Predicting deception from linguistic styles. *Personality and Social Psychology Bulletin*, Vol. 29, No. 5, pp. 665–675.
13. **Ott, M., Choi, Y., Cardie, C., & Hancock, J. T. (2011).** Finding deceptive opinion spam by any stretch of the imagination. *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies-Volume 1*, Association for Computational Linguistics, pp. 309–319.
14. **Pennebaker, J. W., Francis, M. E., & Booth, R. J. (2001).** Linguistic inquiry and word count: LIWC 2001. *Mahway: Lawrence Erlbaum Associates*, Vol. 71, No. 2001, pp. 2001.
15. **Pérez-Rosas, V., Kleinberg, B., Lefevre, A., & Mihalcea, R. (2017).** Automatic detection of fake news. *arXiv preprint arXiv:1708.07104*.
16. **Pérez-Rosas, V., Kleinberg, B., Lefevre, A., & Mihalcea, R. (2018).** Automatic detection of fake news. *Proceedings of the 27th International Conference on Computational Linguistics*, Association for Computational Linguistics, pp. 3391–3401.
17. **Semin, G. R. & Fiedler, K. (1988).** The cognitive functions of linguistic categories in describing persons: Social cognition and language. *Journal of Personality and Social Psychology*, Vol. 54, No. 4, pp. 558.
18. **Senter, R. & Smith, E. A. (1967).** Automated readability index. Technical report, Cincinnati university, OH.
19. **Stone, P. J., Dunphy, D. C., & Smith, M. S. (1966).** *The general inquirer: A computer approach to content analysis*. MIT press.

20. Wang, W. Y. (2017). "Liar, Liar Pants on Fire": A new benchmark dataset for fake news detection. *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, Association for Computational Linguistics, pp. 422–426.
21. Zhang, J., Cui, L., Fu, Y., & Gouza, F. B. (2018). Fake news detection with deep diffusive network model. *arXiv preprint arXiv:1805.08751*.
22. Zhao, Z., Zhao, J., Sano, Y., Levy, O., Takayasu, H., Takayasu, M., Li, D., & Havlin, S. (2018). Fake news propagate differently from real news even at early stages of spreading. *arXiv preprint arXiv:1803.03443*.

Article received on 25/01/2019; accepted on 04/03/2019.  
Corresponding author is Maria Janicka.



# Cochlear Mechanical Models used in Automatic Speech Recognition Tasks

José Luis Oropeza Rodríguez, Sergio Suárez Guerra

Instituto Politécnico Nacional,  
Centro de Investigación en Computación,  
Mexico

{joropeza, ssuarez}@cic.ipn.mx

**Abstract.** In this paper we show that its possible unify two theories that we can find in the state of the art related with human hearing, one of them related with human perceptual phenomenon and the another one related with cochlear mechanic's models linear. The first of them has been used since decade 1980's into Automatic Speech Recognition Systems (ASRs) with satisfactory results. Whereas the second has been used since decade 1950's but never used for ASRs. Since the second is the inner functionality with respect to the first, we propose that is very important to have a study about the behavior of the cochlea models into ASR tasks and compare the results that we can obtain. Then we present an auditory signal processing model that has been proposed as an alternative to the traditional filter banks and LPC models for speech spectral analysis. The argument for such a model is that, because it is based on known properties of the human auditory model (i.e. a model of the cochlea mechanics), it is inherently a better representation of the relevant spectral information that either a traditional bank-filter or an LPC model. In this work we use two different models of the cochlea that they are based in the classic mechanical to analyze their behavior when they are employed for ASR tasks with two variants and two more equations related with the place theory proposed by Von Békésy. Also, we propose an alternative solution for another model based in the fluid mechanical. One time that we analyzed the response of the cochlea with different linear mechanical models we extracted features for ASR tasks that follow the cochlea behavior described by these models. The results obtained demonstrate that our proposal represents a real alternative to be considered for this kind of computational applications. We obtained 2% of higher performance that when we used MFCC parameters in major cases.

**Keywords.** Cochlea, automatic speech recognition, mechanical cochlea models, fluid mechanics, forced harmonic oscillator.

## 1 Introduction

Automatic Speech Recognition Systems, in recent years, have been benefited with the use of the computational model of the auditory periphery. In this paper we propose a new approach that takes some ideas from physiologic model of the cochlea present in state of the art and one of the most important aspects used in the ASR that use modeling of the psychoacoustics, the human perception of the sound with the goal to have a new set of features extracted from the speech signal and used in ASR tasks. The last because the evolution of automatic speech recognition (ASR) points out that employing principle having counterparts in the human auditory system may lead to better performance.

The greatest common denominator of all recognition systems is the signal-processing front end, which converts the speech waveform to some type of parametric representation (generally at a considerably lower information rate) for further analysis and processing.

A wide range of possibilities exists for parametrically representing the speech signal. But principally it exists two dominant methods of spectral analysis, namely, the filter-bank spectrum analysis model, and the linear predictive coding (LPC) spectral analysis model [1].

For a long time, Automatic Speech Recognition Systems have used parameters related with Cepstrum and Homomorphic Analysis of Speech [1, 2] Linear Prediction Coefficients (LPCs) [3], Mel Frequency Cepstrum Coefficients (MFCCs) [4], Perceptual Linear Prediction Coefficients (PLPs) [5], these last two being the most important.

Other tasks where the reduction of the information of the speech signal is relevant are there when a great amount of reference information, such as speech signals for ASR that employed digital networks, is stored. Then, the reduction in the capacity of this information is a problem when we process database speech [1].

Inside the cochlea, a particular frequency analysis is realized. It transforms frequency response into distance response [6]. Then, the solutions before mentioned take only the perceptual response without considering the principal operation of the cochlea.

On the other hand, the most important organ in human hearing is the cochlea and various phenomenological and physiological models have been proposed for a long time. [7, 8, 9]. Cochlear mechanics is a field that relies strongly on fluid mechanics, linear and nonlinear signal processing, and additional mathematical tools. This is applied to a biological structure.

For another side, since 1930's and after 1950's, the analysis and study of the cochlea behavior has generated many publications and considered aspects related with the audition into inner ear that has a capability to divide the sound coming to the outer and medium ear and process the sound that it captures to divide into a set of frequencies.

In these studies, a set of models to represent the operation of the cochlea has been proposed [7, 10, 11, 12, 13, 14, 15].

This paper proposes new parameters, that they are used for ASR tasks, that are related with the fluid mechanical model of the cochlea proposed by Lesser and Berkeley [16] where we propose an alternative solution from the equations gave by them and another based in the macro and micro mechanical model of the cochlea [17, 18], where we used 2 variants find it in the state of the art [19, 20, 21] also we used a proposal related with the cochlea behavior to compare with our results, and another empirical mathematical proposition to analyze the results of our proposal.

Then our hypothesis consider that is possible incorporate a model related with the physiological cochlea model. Now, we are going to review some works that are related with that we mentioned above.

## 2 State of the Art

In [22] the authors proposed a feature extraction method for ASR based on the differential processing strategy of the AVCN, PVCN and the DCN of the nucleus cochlear. The method utilized a zero-crossing with peak amplitudes (ZCPA) auditory model as synchrony detector to discriminate the low frequency formants. They used Continuous density Hidden Markov Models with isolated digits from the Tldigits with 15 states per digit and 5 mixture components per state. A 3-state silence/pause model was inserted at the beginning of each utterance.

They presented a feature extraction for sound data that was motivated by the neural processing of the human auditory system.

The aim of that paper was using generated pulse spiking trains of the auditory nerve fibers that was connected to a feed forward timing artificial Hubel-Wiesel network, which is a structured computational map for higher cognitive functions as e.g. vowel recognition.

The core of the system was a feed-forward timing artificial Hubel-Wiesel network (HW-ANN). Harczos et al. coupled the network with the neural spike output of the ANFs.

The cross-validated recognition rate from segments in the center of the vowels is 68.0%. When neglecting the most confusing vowel (/uw/) which is quite correctly recognized on the training data but loses recognition accuracy on the test data the rate can be further improved up to 85.1%.

In [23], they indicate that hearing has already been modeled up to the cochlear nucleus (CN) to some degree. They used these features without any other spectral information to carry out speech recognition tasks under different noise conditions on the TIMIT database. They found that the shapes of the cochlear delay trajectories carry precious information, which can be extracted even in the presence of noise. This finding may play an important role in next generation cochlear implants.

In this paper we used the same procedure to obtain the Mel-Cepstrum Coefficients because of they have a satisfactory behavior supported by the empirical evidence but the bank of filters distribution is based on cochlear mechanical models.

In this work, as we mentioned above a new set of parameters are obtained from the two models founded in the state of the art related with cochlear mechanics, specifically fluid mechanics and macro and micro mechanic. The corpus SUSAS in English language was used, also Spanish digit corpus, was created. Hidden Markov Models to training and recognition stages, were used. We modify Hidden Markov Model Toolkit to analyze the results obtained with our proposal.

This paper is organized as follows: Cochlea physiology description is introduced briefly specifically describing the pre-processing and processing of the speech signal for the feature extraction are detailed, in Section 2. At same time, section 3 describes our proposal based in two cochlear mechanics models, and we indicate how to obtain the new proposed parameters. Experimental results are described in Section 4, using SUSAS Corpus cleaning. Finally, the conclusions are shown in Section 5.

### 3 Materials and Methods

The ear has three distinct regions called the outer ear, the middle ear, and the inner ear. The outer ear consists of the pinna (the ear surface surrounding the canal in which sound is funneled), and the external canal. Sound waves reach the ear and are guided through the outer ear to the middle ear, which consists of the tympanic membrane or eardrum upon which the sound wave impinges and causes to move and a mechanical transducer (the malleus or hammer, the incus or anvil, and the stapes or stirrup), which converts the acoustical sound wave to mechanical vibrations along the inner ear.

The inner ear consists of the cochlea, which is a fluid-filled chamber partitioned by the basilar membrane, and the cochlea or auditory nerve. The mechanical vibrations impinging on the oval window at the entrance to the cochlea create standing waves (of the fluid inside the cochlea) that cause the basilar membrane to vibrate at frequencies commensurate with the input acoustic wave frequencies (e. g., the formants of voiced speech) and at a place along the basilar membrane that is associated with these frequencies.

The cochlea is a long, narrow, fluid-filled tunnel which spirals through the temporal bone. This tunnel is divided along its length by a cochlear partition into an upper compartment called scala vestibuli (SV) and lower compartment called scala tympani (ST). At the apex of the cochlea, SV and ST are connected to each other by the helicotrema [24]. A set of models to represent the operation of the cochlea has been proposed [7-21]. In mammals, vibrations of the stapes set up a wave with a particular shape on the basilar membrane. The amplitude envelope of the wave is first increasing and then decreasing, and the position at the peak of the envelope is dependent on the frequency of the stimulus [25]. The amplitude of the envelope is a two-dimensional function of distance from the stapes and frequency of stimulation this is that is known as the place theory. The curve shown in Fig. 1 is a cross-section of the function for fixed frequency.

Frequency responses analyzed by Von Békésy are shown in Fig. 1, where each part of the basilar membrane responds maximally to a certain frequency, and as the frequency increases so does the maximum place of the envelope. If low frequencies excite the cochlea, the envelope is nearest to the apex, but if high frequencies excite it, the envelope is nearest to the base.

The displacement pattern of basilar membrane motion is related with high frequencies reaching their apogee towards the base of the cochlea and low frequencies achieving their maximum near the apex (Von Békésy, 1960).

Also, the maintenance of this neural spatial representation of frequency throughout the nuclei of the central pathways is referred to as tonotopic organization.

Now we are going to describe two cochlear mechanical models used in this work, we selected them because they are two of the most important and referenced works in the state of the art of cochlear models, also the results that are obtained with them are nearly to the response that Von Békésy found it in his experiments with corpses.

The first model that we are going to study was proposed by [16] and principally is an equation extracted from the fluid mechanical model to find a relationship between these frequencies and the place of the excitation into the cochlea.

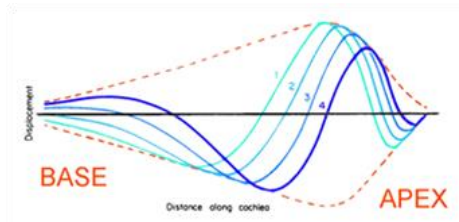


Fig. 1. Wave displacement inside cochlea

With that value a new distribution of the bank filter to extract parameters for ASR tasks is proposed.

Let  $u = (u_1, u_2, u_3)$  be the fluid velocity,  $p$  the pressure, and  $\rho$  the constant density of the fluid. The mass of fluid in a fixed volume  $V$  can change only in response to fluid flux across the boundary of the volume. Thus according [24, 16]:

$$\frac{d}{dt} \int_V \rho dV = - \int_S \rho(u \cdot n) dS = 0, \quad (1)$$

where  $S$  is the surface of  $V$ , and  $n = (n_1, n_2, n_3)$  is the outward unit normal to  $V$ .

After considering that the momentum of the fluid in a fixed domain  $V$  can change only in response to applied forces or to the momentum flux across the domain boundary, and using the divergence theorem to convert surface integrals to volume integrals, 2 is obtained:

$$\int_V \left( \rho \frac{\partial u_i}{\partial t} + \rho \nabla \cdot (u_i u) + \frac{\partial p}{\partial x_i} \right) dV = 0. \quad (2)$$

After considering that  $V$  is arbitrary, fluid motions are of small amplitude and there is an irrotational flow, the following equations are shown:

$$\begin{aligned} \rho \frac{\partial \phi}{\partial t} + p &= 0, \\ \nabla^2 \phi &= 0 \end{aligned} \quad (3)$$

Lesser and Berkley developed a model that combines these last two equations with the equation of a damped, forced harmonic oscillator and is considered one of the simplest of the cochlea models.

They proposed that each point of the basilar membrane is modeled as a simple damped harmonic oscillator with mass, damping, and stiffness that vary along the length of the membrane. Thus, the movement of any part of the membrane is assumed to be independent of the movement of neighboring parts of the membrane, as there is no direct lateral coupling. The deflection of the basilar membrane,  $\eta(x, t)$ , is specified by a model of a forced harmonic oscillator defined as:

$$m(x) \frac{\partial^2 \eta}{\partial t^2} + r(x) \frac{\partial \eta}{\partial t} + k(x) \eta = p_2(x, \eta(x, t), t) - p_1(x, \eta(x, t), t), \quad (4)$$

where  $m$  is the mass,  $R_m$  mechanical resistance and  $k$  is the damping constant which can be substituted by following values  $m(x) = 0.1$ ,  $r(x) = 300e^{-ax}$ ,  $k(x) = 10^9 e^{-2ax}$ . An analytical solution of this problem can be found using standard Fourier series [16]. Solutions of this form are looked for:

$$\phi = x \left( 1 - \frac{x}{2} \right) - \sigma y \left( 1 - \frac{y}{2\sigma} \right) + \sum_{n=0}^{\infty} A_n \cosh[n\pi(\sigma - y)] \cos(n\pi x). \quad (5)$$

## 4 Auditory Models

This paper proposes solving the Lesser and Berkley equation using the solution proposed in [26]. This solution is related with the place theory of hearing, initially proposed by Von Békésy. To perform the analysis each section of the membrane is considered as a forced harmonic isolated oscillator, which is excited by an external force that represents the driving force on each section of the basilar membrane and this force is produced by vibrations transmitted into the cochlea by the oval window. Two solutions are proposed related with the before mentioned equation. Firstly, the forced harmonic oscillator is represented by the following equation:

$$m(x) \frac{d^2 \eta}{dt^2} + R_m(x) \frac{d\eta}{dt} + k(x) \eta = F e^{j\omega t}, \quad (6)$$

where  $m$  is the mass,  $R_m$  mechanical resistance and  $k$  is the damping constant. Considering that  $\eta = A e^{j\omega t}$ , then amplitude of the wave sound into the cochlea is represented by [26]. Secondly, a



damped harmonic oscillator with the following equation is considered:

$$m(x)\frac{d^2\eta}{dt^2} + R_m(x)\frac{d\eta}{dt} + k(x)\eta = 0 \quad (7)$$

then, a solution is given by:

$$\eta = Ae^{-\beta t} \cos(\omega_0 t + \phi) \quad (8)$$

Equation 8 shows that the amplitude for each section of the membrane depends of the frequency  $\omega$  in the applied force. The amplitude has a maximum when the denominator has its minimum value and this occurs at a specific frequency excitation called resonance frequency.

This is defined by the values of mass and stiffness, when the frequency  $\omega$  of the applied force is equal to  $k(x)/m(x)$  it is said that the system is resonant in amplitude and obtains the maximum value of the basilar membrane displacement.

This last equation can be expressed as a function of frequency and distance, if considering that  $\omega = 2\pi f$  thus, this is possible using our purpose:

$$A = \frac{F/m(x)}{\sqrt{\left(4\pi^2 f^2 - \frac{k(x)}{m(x)}\right)^2 + 4\pi^2 f^2 \frac{R_m(x)^2}{m(x)^2}}} \quad (9)$$

In the literature we can find another two equations that they intent to represent the behavior of the cochlea. One of them proposed by Greenwood [27] that is represented in the equation 10. The generalized form of the mammalian reception cochlear map which is described by the Greenwood relation:

$$f = A(10^{ax} - k), \quad (10)$$

where  $f$  is frequency (Hz),  $x$  is the distance from the apex of the cochlea (helicotrema end),  $A$ ,  $a$ , and  $k$  are coefficients [28, 29, 30],  $a$  is the gradient of high frequency end of the map, i.e., the coefficient of the derivative evaluated at the highest frequency,  $A$  is a constant which shifts the curve as a whole along the log-frequency axis, and  $k$  is constant, which introduces curvature into the frequency position function so as to fit low-frequency data.

And another called empirical equation of the cochlea behavior that is represented in the equation 11 [31]:

$$f = 10^{4-1.5\tan(x/3)}. \quad (11)$$

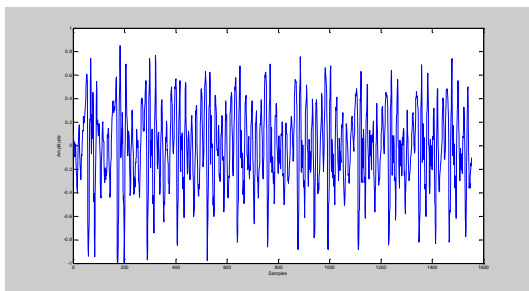
The following figures 2(a-k) illustrate different aspects related with a segment of the speech signal analysis and the information extracted from them.

Figure 2a) shows a segment of the speech signal that we analyze, 2b) shows the spectral representation of the segment of the speech signal, 2c) shows the spectral envelope of the spectral representation extracted from the LPC analysis, 2d) shows the spectrogram of the segment of the speech signal, 2e) shows the behavior of the relation between distance vs excitation frequency founded by our proposal and represented by the equation 9 is reached, 2f) shows the relation between resonance frequency and frequency of the excitation of our proposal, 2g) shows the relation of the amplitude vs frequency of the excitation for our proposal, 2h) shows a bank of triangular filters constructed from our propose, 2i) shows the values of the parameters obtained from our proposal, and 2j) and 2k) show the spectral response and the spectral representation of the speech signal after that has been passed for the bank of filters constructed from our proposal, respectively.

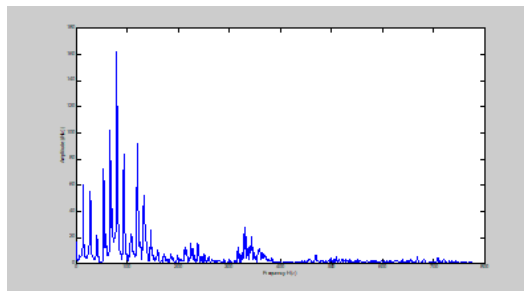
The second model that we used to came on of an equation extracted from a mechanical model to find a relationship between these frequencies and the place of the excitation into the cochlea.

With that value a new distribution of the bank filter to extract parameters for ASR tasks is proposed.

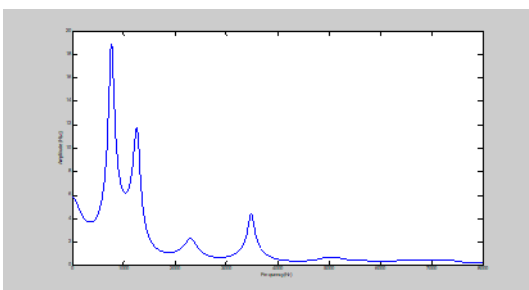
For that the micromechanical the anatomical structure of a radial cross-section (RCS) of the cochlear partition (CP) is illustrated in the following figure 3. In the model, the basilar membrane (BM) and tectorial membrane (TM) are each represented as a lumped mass with both stiffness and damping in their attachment to the surrounding bone.



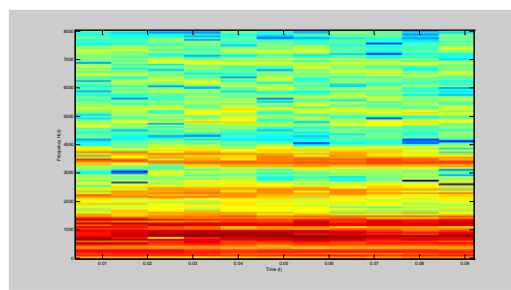
**Fig. 2a).** Shows a segment of the speech signal that we analyze



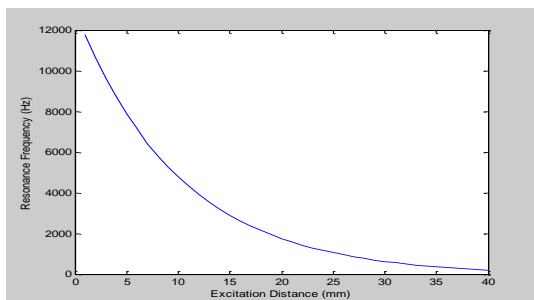
**Fig. 2b).** Shows the spectral representation of the segment of the speech



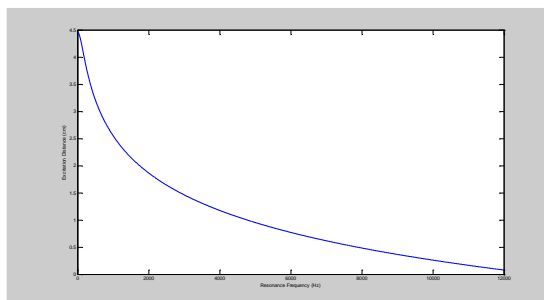
**Fig. 2c).** Shows the spectral envelope of the spectral representation extracted from the LPC analysis



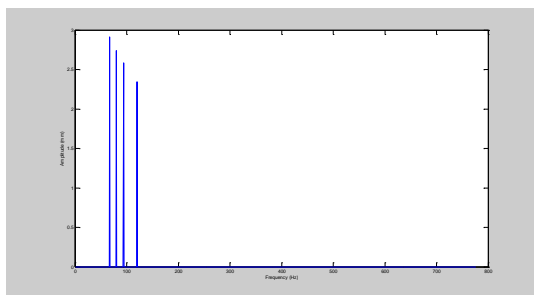
**Fig. 2d).** Shows the spectrogram of the segment of the speech



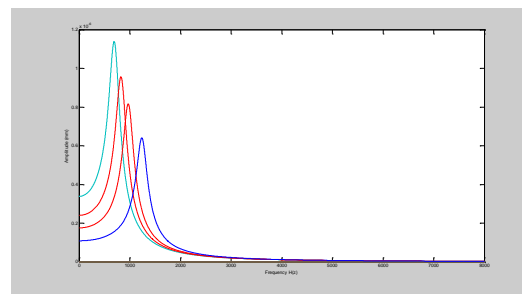
**Fig. 2e).** Shows the behavior of the relation between distance vs excitation frequency founded by our proposal and represented by the equation 9 is reached



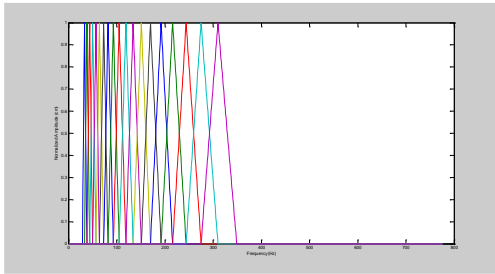
**Fig. 2f).** Shows the relation between resonance frequency and frequency of the excitation of our proposal



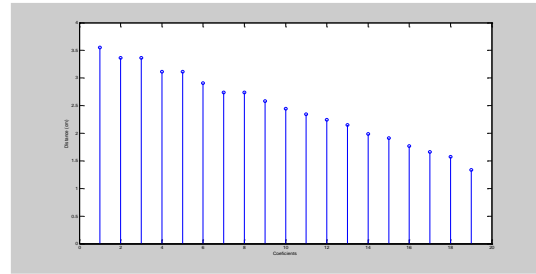
**Fig. 2g)** Shows the relation of the amplitude vs frequency of the excitation for our proposal



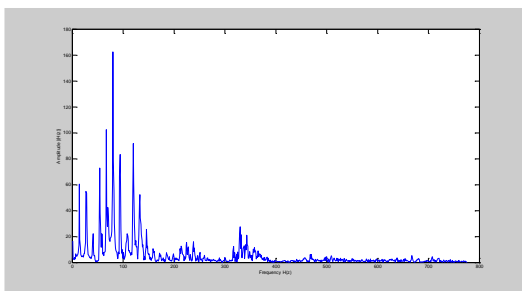
**Fig. 2h).** Shows the relation of the amplitude vs frequency of the excitation for our proposal



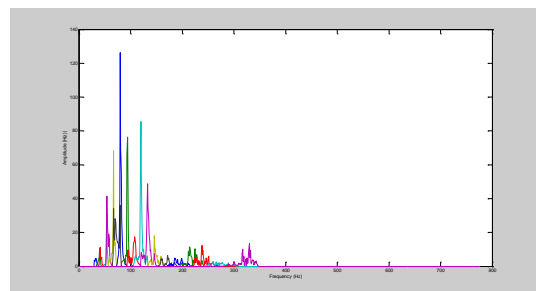
**Fig. 2i).** Shows a bank of triangular filters constructed from our proposal



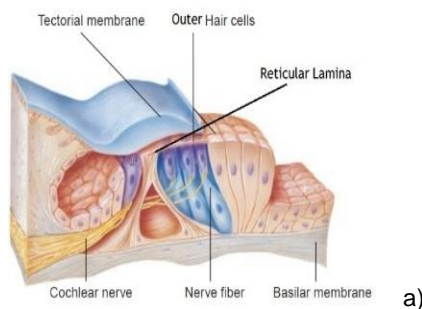
**Fig. 2j).** Shows a bank of triangular filters constructed from our proposal



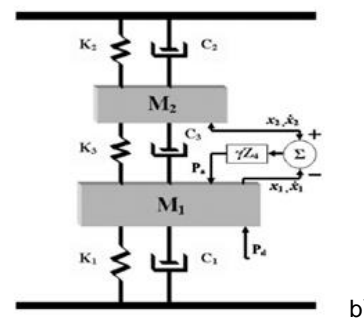
**Fig. 2k).** Spectral response



**Fig. 2l).** Spectral response after that the spectral representation of the speech signal has been passed for the bank of filters constructed from our proposal



a)



b)

**Fig. 3.** (a) Anatomical structure of the cochlear partition, (b) The outer hair cells, micro mechanical representation

When the cochlea determines the frequency of the incoming signal from the place on the basilar membrane of maximum amplitude, the organ of Corti is excited, in conjunction with the movement of tectorial membrane; the inner and outer hair cells are excited obtaining an electrical pulse that travels by auditory nerve.

Now the modeling cochlear will be divided in two ways of study. The first is the hydrodynamic

movement that produced a movement on the basilar membrane and the second is the movement of the outer hair cells. This is named as the model of Macro and Micro Mechanical Cochlear [18]. The equations that describe the Macro Mechanical Cochlear are [18]:

$$\frac{d^2}{dx^2} P_d(x) = \frac{2\rho}{H} \varepsilon(x), \quad (12)$$

**Table 1.** Values used in equation

Parameters	Neely & Kim 1986 (cgs)	Ku (human cochlea, 2008)	Elliot (2007)
$k_1(x)$	$1.1 * 10^9 e^{-4x}$	$1.65 * 10^8 e^{-2.79(x+0.00373)}$	$4.95 * 10^8 e^{-3.2(x+0.00375)}$
$c_1(x)$	$20 + 1500 e^{-2x}$	$0.9 + 999 e^{-1.53(x+0.00373)}$	$0.1 + 1970 e^{-1.79(x+0.00375)}$
$m_1(x)$	$3 * 10^{-3}$	$4.5 * 10^{-4}$	$1.35 * 10^{-3}$
$k_2(x)$	$7 * 10^6 e^{-4.4x}$	$1.05 * 10^6 e^{-3.07(x+0.00373)}$	$3.15 * 10^6 e^{-3.52(x+0.00375)}$
$c_2(x)$	$10 e^{-2.2x}$	$3 e^{-1.71(x+0.00373)}$	$11.3 e^{-1.76(x+0.00375)}$
$m_2(x)$	$0.5 * 10^{-3}$	$0.72 * 10^{-4} + 0.28710^{-2}x$	$2.3 * 10^{-4}$
$k_3(x)$	$1 * 10^7 e^{-4x}$	$1.5 * 10^6 e^{-2.79(x+0.00373)}$	$4.5 * 10^6 e^{-3.2(x+0.00375)}$
$c_3(x)$	$2 e^{-0.8x}$	$0.66 e^{-0.593(x+0.00373)}$	$2.25 e^{-0.64(x+0.00375)}$
$k_4(x)$	$6.15 * 10^8 e^{-4x}$	$9.23 * 10^7 e^{-2.79(x+0.00373)}$	$2.84 * 10^8 e^{-3.2(x+0.00375)}$
$c_4(x)$	$1040 e^{-2x}$	$330 e^{-1.44(x+0.00373)}$	$965 e^{-1.64(x+0.00375)}$
gamma	1	1	1
g	1	1	1
b	0.4	0.4	0.4
L	2.5	3.5	3.5
H	0.1	0.1	0.1
$K_m$	$2.1 * 10^6$	$2.63 * 10^7$	$2.63 * 10^7$
$C_m$	400	$2.8 * 10^3$	$2.8 * 10^3$
$M_m$	$45 * 10^3$	$2.96 * 10^{-3}$	$2.96 * 10^{-3}$
$C_h$	0.1	35	21
$A_s$	0.01	$3.2 * 10^{-2}$	$3.2 * 10^{-2}$
Rho	0.35	1	1
N	250	500	500
Gm	0.5	0.5	0.5

$$\frac{d^2}{dx^2} P_d(0) = 2\rho \ddot{\varepsilon}_s \quad (13)$$

$$\frac{d^2}{dx^2} P_d(L) = 2\rho \ddot{\varepsilon}_h \quad (14)$$

The equations (12, 13, 14) were solved by finite difference, using central differences for (12), forward differences for the (13) and backward difference for (14), generating a tridiagonal Matrix system [32] which we solved using the Thomas algorithm.

It represents the Micro mechanical, because it uses the organ of Corti values. The solution for  $P_d$  obtains the maximum amplitude on the basilar membrane shown in Figure 1. For these experiments the cochlear distance pattern is obtained manually. As can be seen, to solve equation 15 a set of variables related with the physiology of the cochlea is needed and some of these variables are described in table 1.

These values are immersed into  $Z_p$  and  $Z_m$ ; for example in [18]. One aspect important to mention is that the values showed in table 1 are values of the human body and some of them are not the

**Table 2.** Frequency vs. proposed distance

Our proposal			Neely-Elliot		Neely-Ku		Empirical	
Item	frequency	distance (x)	frequency	distance (x)	frequency	distance (x)	frequency	distance (x)
1	78.825638	4.33948	209	3.471943888	361	3.240480962	299.9296166	2.752
2	116.197052	4.226435	254	3.396348252	395	3.1750167	346.4698275	2.690148148
3	150.462677	4.11339	303	3.320752616	438	3.109552438	397.9246918	2.628296296
4	184.646286	4.000345	338	3.245156981	490	3.044088176	454.5940071	2.566444444
5	220.126266	3.8873	385	3.169561345	542	2.978623914	516.7886269	2.504592593
6	257.798676	3.774255	431	3.093965709	597	2.913159653	584.8319308	2.442740741
7	298.382843	3.66121	501	3.018370073	663	2.847695391	659.0614242	2.380888889
8	342.535828	3.548165	545	2.942774438	728	2.782231129	739.8304708	2.319037037
9	390.906982	3.43512	606	2.867178802	807	2.716766867	827.5101673	2.257185185
10	444.168732	3.322075	683	2.791583166	884	2.651302605	920.9245991	2.195333333
11	503.037781	3.20903	763	2.715987531	968	2.585838343	1023.494306	2.133481481
12	568.290527	3.095985	843	2.640391895	1058	2.520374081	1134.208285	2.07162963
13	640.777527	2.98294	939	2.564796259	1168	2.45490982	1253.530073	2.009777778
14	721.436951	2.869896	1045	2.489200623	1277	2.389445558	1381.952813	1.947925926
15	811.307861	2.756851	1152	2.413604988	1407	2.323981296	1520.002279	1.886074074
16	911.545593	2.643806	1292	2.338009352	1536	2.258517034	1668.240157	1.824222222
17	1023.43585	2.530761	1422	2.262413716	1677	2.193052772	1827.267638	1.76237037
18	1148.4126	2.417716	1581	2.186818081	1847	2.12758851	1997.729347	1.700518519
19	1288.07617	2.304671	1741	2.111222445	2015	2.062124248	2177.320929	1.638666667
20	1444.21497	2.191626	1935	2.035626809	2198	1.996659987	2372.570236	1.576814815
21	1618.82715	2.078581	2151	1.960031173	2398	1.931195725	2584.911035	1.514962963
22	1814.1471	1.965536	2392	1.884435538	2639	1.865731463	2804.915918	1.453111111
23	2032.67358	1.852491	2663	1.808839902	2879	1.800267201	3043.811772	1.391259259
24	2277.20313	1.739446	2935	1.733244266	3139	1.734802939	3299.179135	1.329407407
25	2550.86353	1.626401	3269	1.657648631	3457	1.669338677	3572.113841	1.267555556
26	2857.15649	1.513356	3643	1.582052995	3770	1.603874415	3863.804939	1.205703704
27	3200	1.400312	4061	1.506457359	4113	1.538410154	4170.432437	1.143851852

same with respect at Neely, Ku or Elliot used in their papers. The most important values are obtained from [33].

One important aspect to indicate is that before 300 Hz the behavior of the micro and macro mechanical model is not adequate, independently of the parameters used. This result is a consequence of the characteristics of the model proposed by [18].

Proposing our analysis from this frequency to 4.5 KHz was decided. Also, the response obtained has a behavior logarithmic.

This is an important indication because the Mel function is related with a similar mathematical function.

One of the most important aspects related with the cochlear models is that the equations when they are substituted with adequate values the response of the system must to be same at the Von

Békésy proposed in his research about the human cochlea. At same time, we can see that the pressure to reach a maximum in a value of distance inside of the cochlea. This value of distance from the apex to the helicotrema is the value that we can use to obtain the feature from our purpose that we are going to use for the next set of experiments.

## 5 Experiments and Results

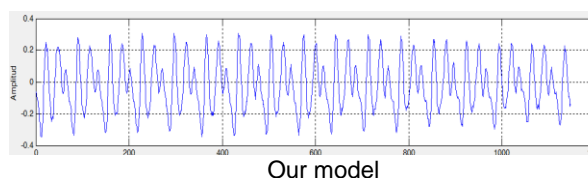
One time that we have the analysis of the micro and macro mechanical model of the cochlea with the last equations we can obtain a mathematical expression for the behavior of the distance vs frequency of the excitation such as we mentioned for the Lesser and Berckley model, or we can evaluate the response of the cochlea with different values of the frequency of the excitation and then to find the distance  $x$  where we can obtain a maximum, such as last figure shows.

Then now we have a model proposed by Neely with three different values, each of one them with their properties. Table 2 shows the values for Neely values. Neely-Ku, Neely-Elliott use the same model developed by Neely but with different values and analysis [21].

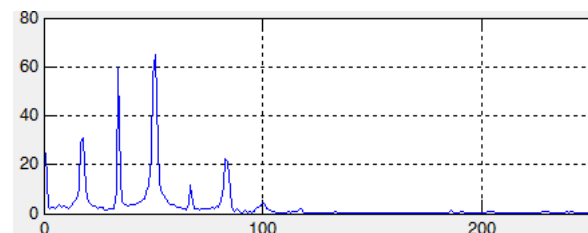
An important aspect that we can see is that independently of the model that we used the behavior of the cochlea follows the same pattern, that is the curve of the response, is not linear and approximately logarithmic. From this response we can conclude that psychoacoustic response has reflected the behavior that we can observe in the cochlea as the models show, that is the Mel scale or Bark scale have their causes because they follow the cochlear response.

As it was hoped then the outer ear response depends almost of the inner ear behavior, and middle ear has a little repercussion in the speech analysis because of it works as an impedance coupler between outer ear and inner ear.

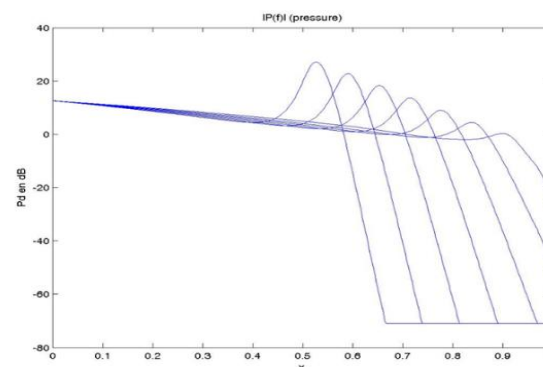
The last situation was the principle aspect that we use to indicate that we can obtain a set of parameters from the cochlea behavior as we use features as MFCC or PLP for Automatic Speech Recognition tasks with the difference that the behavior of the cochlea is nearer to the response of the nervous system of the human body.



**Fig. 4.** One speech signal segment and spectral representations



**Fig. 5.** Spectral representation of one segment of speech signal after that they are processed by the triangular bank filters proposed in this work



**Fig. 6.** Response curve found using second model

The next figure 5 represents spectral representation of one segment of the speech signal. These spectral representations were obtained after of the speech signal was passed by triangular bank filters constructed from our proposal. At this moment we have a set of mechanical models of the human cochlea that describe how the sound signal affects to the basilar membrane.

We must remember that the movement of the basilar membrane must excite to a set of cilia cells that send an electric signal to the hearing nervous that send this signal to nucleus cochlear.

---

**Algorithm 1. Steps associated to the new speech parameters proposed in this paper.**


---

1. Obtain speech signal, realize preprocessing (It includes pre-emphasis, segmentation, windowing and feature extraction), for each sentence.

2. In the feature extraction, the same procedure as MFCC was used but the filter bank is constructed following the next steps.

2.1 Take the minimal and maximal frequency where filter bank are going to be constructed.

2.2 Calculate maximal and minimal distance from the stapes of the cochlea, nearer to start implies high frequencies, farthest implies low frequencies.

2.3 Determine a set of distances equally spaced

3. Determine the frequency related with these distances, this represents the center of the filter bank.

4. Construct filter bank with frequency center obtained from the analysis of the Neely model using values in table

5. Follow the same steps to obtain MFCC, multiply spectral representation from Fourier Transform with filter bank, calculate energy by bands using logarithm, and finally, apply discrete cosine transform.

6. Obtain a new set of coefficients for each speech signal.

6.1 Train the ASR and proceed with recognition task using the new parameters.

---

One of the most important aspects that we can see from the curves is that the values used for the frequency causes a specific excitation in a specific distance into the cochlea, this aspect follows the response that Von Békésy mentioned in his place theory. From this response of the cochlea we could propose that a filter bank with a triangular form is adequate to analyze this behavior, because the response is punctual as show figure of the presion in the basilar membrane.

For this work we propose that triangular bank filter is accepted aunque another important design can be used. Then now we have enough elements for to have a new set of parameters based on the analysis before described.

Next, we are going to explain how we can obtain a set of parameters for ASRs tasks from these ideas. As we mentioned above, the Neely model and later works have considered putting a number of these micro-mechanisms along the cochlea at the same distance between them. For that, this principle to establish the following relation between a minimal and maximal distance was use:

$$d(n) = d_{\max} + \sum_{n=0}^{n_{\text{int}}} n \frac{d_{\min} - d_{\max}}{n_{\text{int}} + 1} \quad (16)$$

In 5  $d_{\min}$  and  $d_{\max}$  are obtained from Figures 5 for each case, considering that  $F_{\min}=300$  Hz and  $F_{\max}=4.5$  KHz. This paper proposed a space equidistant between different points to analyze the cochlea. After that, for each distance one specifically frequency of excitation to the Basilar Membrane was obtained. Figure 5 shows this behavior.

From the last analysis a computational model to obtain the distance, where the maximum displacement of the basilar membrane occurs to a specific excitation frequency of the system was developed, which depends of the physical characteristics of the basilar membrane. The following procedure describes the computational model of the cochlea using this proposal [20]. It is important to mention that the maximum response of the pressure curve used in [19] was obtained.

The first experimental used a database that contains only digits in the Spanish language and the characteristics of the samples were frequency sample 11025, 8 bits per sample, PCM coding, mono-stereo.

The evaluation of the experiment proposed involved 5 people (3 men and 2 women) with 300 speech sentences to recognize for each one ( 100 for training task and 200 for recognition task). 1500 speech sentences extracted from 5 speakers individually were taken, and the Automatic Speech Recognition trained using Hidden Markov Models with 6 states (4 states with information and 2 dummies to connection with another chain). Also, 3 Gaussian Mixture for each state in the Markov chain were employed.

The parameters extracted from the speech signal were 39 (13 MFCC, 13 delta and 13 energy coefficients) when using MFCC or our proposal, and used to train the Hidden Markov Model.

**Table 3.** LPC, CLPC, MFCC and delta; acceleration, delta, and third differential coefficients

SENTENCES				WORDS			
PARAM/# STATE	4	5	6	PARAM/#STATE	4	5	6
LPC	77	89.5	9	LPC	77.39	89.95	89.45
CLPC	89.5	99	9	CLPC	89.95	99.5	99.5
MFCC	98.5	99	9	MFCC	98.99	99.5	99.5
CMCC KU	100	100	100	CMCC KU	100	100	100
CMCC ELLIOT	100	100	100	CMCC ELLIOT	100	100	100
CMCC NEELY	100	100	100	CMCC NEELY	100	100	100
CMCC RESONAN	99.4	99.6	99.8	CMCC RESONAN	99.6	99.8	99.8

**Table 4.** Results obtained using HTK, SUSAS Corpus and automatic labeling

boston 1										
	SENTENCE	WORD	H	S	N	H	D	S	I	N
CMCC_Elliot	90.2	90.48	221	24	245	228	7	17	0	252
CMCC_Empirico	93.06	93.25	228	17	245	235	7	10	0	252
CMCC_Greenwood	93.88	94.05	230	15	245	237	7	8	0	252
CMCC_Ku	92.65	92.86	227	18	245	234	7	11	0	252
CMCC_L&B_RA	90.2	90.48	221	24	245	228	7	17	0	252
MFCC	91.84	92.06	225	20	245	232	7	13	0	252
CMCC_Neely	91.43	91.67	224	21	245	231	7	14	0	252
boston 2										
	SENTENCE	WORD	H	S	N	H	D	S	I	N
CMCC_Elliot	94.69	94.84	232	13	245	239	7	6	0	252
CMCC_Empirico	94.29	94.44	232	13	245	239	7	6	0	252
CMCC_Greenwood	94.69	94.84	232	13	245	239	7	6	0	252
CMCC_Ku	95.51	95.63	234	11	245	241	7	4	0	252
CMCC_L&B_RA	93.88	94.05	230	15	245	237	7	8	0	252
MFCC	95.1	95.24	234	11	245	241	7	4	0	252
CMCC_Neely	93.47	93.65	230	15	245	237	7	8	0	252
boston 3										
	SENTENCE	WORD	H	S	N	H	D	S	I	N
CMCC_Elliot	93.47	93.65	229	16	245	236	7	9	0	252
CMCC_Empirico	96.33	96.43	236	9	245	243	7	2	0	252
CMCC_Greenwood	96.73	96.83	237	8	245	244	7	1	0	252
CMCC_Ku	95.92	96.03	235	10	245	242	7	3	0	252
CMCC_L&B_RA	92.65	92.86	227	18	245	234	7	11	0	252
MFCC	96.73	96.83	237	8	245	244	7	1	0	252
CMCC_Neely	96.73	96.83	237	8	245	244	7	1	0	252
general 1										
	SENTENCE	WORD	H	S	N	H	D	S	I	N
CMCC_Elliot	97.14	96.83	238	7	245	244	7	1	0	252
CMCC_Empirico	95.51	95.63	234	11	245	241	7	4	0	252
CMCC_Greenwood	96.73	96.43	237	8	245	243	7	2	0	252
CMCC_Ku	96.73	96.83	237	8	245	244	7	1	0	252
CMCC_L&B_RA	95.51	95.24	234	11	245	240	7	5	0	252
MFCC	96.73	96.83	237	8	245	244	7	1	0	252
CMCC_Neely	96.33	96.43	236	9	245	243	7	2	0	252



general 2										
	SENTENCE	WORD	H	S	N	H	D	S	I	N
CMCC_Elliot	96.33	<b>96.43</b>	236	9	245	243	7	2	0	252
CMCC_Empirico	95.92	96.03	235	10	245	242	7	3	0	252
CMCC_Greenwood	95.92	96.03	235	10	245	242	7	3	0	252
CMCC_Ku	95.1	95.24	233	12	245	240	7	5	0	252
CMCC_L&B_RA	93.06	93.25	228	17	245	235	7	10	0	252
MFCC	94.29	94.44	231	14	245	238	7	7	0	252
CMCC_Neely	94.29	94.44	231	14	245	238	7	7	0	252
general 3										
	SENTENCE	WORD	H	S	N	H	D	S	I	N
CMCC_Elliot	94.29	94.44	231	14	245	238	7	7	0	252
CMCC_Empirico	93.88	94.05	230	15	245	237	7	8	0	252
CMCC_Greenwood	93.88	94.05	230	15	245	237	7	8	0	252
CMCC_Ku	93.06	93.25	228	17	245	235	7	10	0	252
CMCC_L&B_RA	94.69	94.84	232	13	245	239	7	6	0	252
CMCC_MFCC	93.47	93.65	229	16	245	236	7	9	0	252
CMCC_Neely	95.10	<b>95.24</b>	232	13	245	239	7	6	0	252
nyc1										
	SENTENCE	WORD	H	S	N	H	D	S	I	N
CMCC_Elliot	92.24	92.06	226	19	245	232	7	13	0	252
CMCC_Empirico	91.84	92.06	225	20	245	232	7	13	0	252
CMCC_Greenwood	91.02	91.27	223	22	245	230	7	15	0	252
CMCC_Ku	90.61	90.48	222	23	245	228	7	17	0	252
CMCC_L&B_RA	93.06	<b>92.86</b>	228	17	245	234	7	11	0	252
MFCC	91.84	92.06	225	20	245	232	7	13	0	252
CMCC_Neely	92.24	92.06	226	19	245	232	7	13	0	252
nyc2										
	SENTENCE	WORD	H	S	N	H	D	S	I	N
CMCC_Elliot	94.29	<b>94.44</b>	231	14	245	238	7	7	0	252
CMCC_Empirico	93.47	93.65	229	16	245	236	7	9	0	252
CMCC_Greenwood	93.88	94.05	230	15	245	237	7	8	0	252
CMCC_Ku	91.84	92.06	225	20	245	232	7	13	0	252
CMCC_L&B_RA	89.8	90.08	220	25	245	227	7	18	0	252
MFCC	91.02	91.27	223	22	245	230	7	15	0	252
CMCC_Neely	89.39	89.68	219	26	245	226	7	19	0	252
nyc3										
	SENTENCE	WORD	H	S	N	H	D	S	I	N
CMCC_Elliot	95.1	<b>95.24</b>	234	11	245	241	7	4	0	252
CMCC_Empirico	93.88	94.05	232	13	245	239	7	6	0	252
CMCC_Greenwood	93.47	93.65	229	16	245	236	7	9	0	252
CMCC_Ku	93.06	93.25	229	16	245	236	7	9	0	252
CMCC_L&B_RA	89.39	89.68	221	24	245	228	7	17	0	252
MFCC	94.69	94.84	242	10	245	242	7	3	0	252
CMCC_Neely	94.29	94.44	234	11	245	241	7	4	0	252

It is important to mention that HTK give us results in two forms: by sentence and by words <http://htk.eng.cam.ac.uk>. We show both for reasons of consistency.

Table 3 contains results obtained in percentage when using LPC, CLPC and MFCC, DELTA, ACCELERATION AND THIRD DIFFERENTIAL. We can see clearly that MFCC giving us a good

performance with respect LPC or CLPC parameters, then we obviously used these parameters to compare with our proposal.

In the second experiment, a corpus elaborated by J. Hansen at the University of Colorado Boulder was used. He has constructed database SUSAS (Speech Under Simulated and Actual Stress) <http://catalog ldc.upenn.edu/LDC99S78>. Only 9

speakers with ages ranging from 22 to 76 were used and we applied normal corpus not under Stress sentences contained into corpus.

The words were “brake, change, degree, destination, east, eight, eighty, enter, fifty, fix, freeze, gain, go, hello, help, histogram, hot, mark, nav, no, oh, on, out, point, six, south, stand, steer, strafe, ten, thirty, three, white, wide, & zero”.

A total of 4410 files of speech were processed. Finally, Tables 4 shows results when using our proposal (Cochlear Mechanics Cepstrum Coefficients –CMCC-) the best representations used in the state of the art and in the last experiment versus MFCC in SUSAS corpus. As we can see a new form to obtain feature for ASRs tasks is better with respect traditionally MFCC.

Then we demonstrate that if we use CMCC (Cochlear Mechanics Cepstrum Coefficients) is an interesting alternative in this research area.

## 6 Conclusions and Future Work

This paper describes new parameters for ASRs tasks. They employ the functionality of the cochlea, the most important hearing organ of humans and mammals. At this moment, the parameters used for the MFCC analysis have been demonstrated to be the most important parameters and the most used for this task.

The interest of this paper is show the implementation of the cochlear models in Automatic Speech Recognition tasks. We show that the theory of these models can be used to obtain parameters from the speech signal and used as input to the Hidden Markov Model Toolkit. Also, the paper showed an analytic solution to the Lesser & Berkley model (this model was proposed in 1972 and is based in the mechanical fluid and its solution used the Fourier series), that is based in the resonance analysis proposed by Helmholtz. After that we show a mathematical expression can be compared with another used in the State of the Art, for example the equation of Greenwood and another obtained empirically.

Also we used mechanical model of the cochlea proposed by Neely named micro and macro mechanical, for that we solved the equation system of the model and we determinate the frequency of excitation into the human cochlea for two variants

of this model that exists in the state of the art of cochlear mechanics linear that use the same operating principle.

This article demonstrated that our proposal is very interesting because the performance reached was adequate and can be used to obtain speech signal parameters for Automatic Speech Recognition.

In conclusion, the cochlea behavior can be used to obtain these parameters and the results are adequate. Another aspect to consider for future work is obtain an equation to extract the frequency place relation in model Neely's. And to have an analysis of noise to compare the results when this aspect is add to speech signal and to see in real situations how the results are obtained.

## Acknowledgment

We want to acknowledge The National Polytechnic Institute (IPN, Mexico), Center for Computing Research and especially SIP project number 20181550 for their support.

## References

1. **Rabiner, L. & Juang, B.H. (1993).** Fundamentals of Speech Recognition, Prentice Hall.
2. **Noll, A.M. (1964).** Shorttime Spectrum and Cepstrum Techniques for Vocal Pitch Detection. *Journal of Acoustical Society of America*, Vol. 36, No. 2, pp. 296–302. DOI: 10.1121/1.1918949.
3. **Makhoul, J. (1975).** Linear Prediction: A Tutorial Review. *Proceedings of the IEEE*, Vol. 63, No. 4, pp. 561–580. DOI: 10.1109/PROC.1975.9792.
4. **Davis, S.B. & Mermelstein, P. (1980).** Comparison of Parametric Representations for Monosyllabic Word Recognition in Continuously Spoken Sentence. *Processing IEEE, Transactions on Acoustics*, Vol. 28, No. 4, pp. 357–366. DOI: 10.1109/TASSP.1980.1163420.
5. **Hermansky, H. (1990).** Perceptual Linear Predictive (PLP) analysis of speech. *Journal of Acoustical Society of America*, Vol. 87, No. 4, pp. 1738–1752. DOI: 10.1121/1.399423.
6. **Von-Békésy, G. (1961).** Concerning the pleasures of observing, and the mechanics of the inner ear. *Nobel Lecture*.
7. **Duifhuis, H. (2012).** *Cochlear Mechanics Introduction to a Time Domain Analysis of the*

- Nonlinear Cochlea*. Faculty of Mathematics and Natural Sciences, University of Groningen Nijenborgh 9.
8. Dallos, P., Popper, A.N., & Fay, R.R. (1996). The cochlea. Chapter 5: Mechanics of the cochlea: modeling effects. Vol. 8. DOI: 10.1007/978-1-4612-0757-3.
  9. Robles, L. & Ruggero, M.A. (2001). Mechanics of the Mammalian Cochlea. *Physiological Reviews* Vol. 81, No. 3, pp. 1305–1352. DOI: 10.1152/physrev.2001.81.3.1305.
  10. de Boer, E. (1980). Auditory physics. Physical principles in hearing I. *Physics Reports*, Vol. 62, No. 2, pp. 87–174. DOI: 10.1016/0370-1573 (80) 90100-3.
  11. de Boer, E. (1984). Auditory physics. Physical principles in hearing II. *Physics Reports*, Vol. 105, No. 3, pp. 141–226. DOI: 10.1016/0370-1573(84)90108-X.
  12. de Boer, E. (1991). Auditory physics. Physical principles in hearing II, *Physics Reports*, Vol. 203, No. 3, pp. 125–231. DOI: 10.1016/0370-1573(91)90068-W.
  13. Peterson, L.C. & Bogert, B.P. (1950). A dynamical theory of the cochlea. *Journal of the Acoustical Society of America*, Vol. 22, No. 3, pp. 369–381. DOI: 10.1121/1.1906615.
  14. Zwislocki, J. (1953). Review of Recent Mathematical Theories of Cochlear Dynamics. *Journal of Acoustical Society of America*, Vol. 25, No. 4, pp. 743–751. DOI: 10.1121/1.1907170.
  15. Lesser, M.B. & Berkley, D.A. (1972). Fluid mechanics of the cochlea, *Journal Fluid Mechanics*, Vol. 51, No. 3, pp. 497–512. DOI: 10.1017/S0022112072002320.
  16. Neely, S.T. (1981). Finite difference solution of a two-dimensional mathematical model of the cochlea. *Journal of Acoustical Society of America*, Vol. 69, No. 5, pp. 1386–1396. DOI: 10.1121/1.385820.
  17. Neely, S.T. (1986). A model for active elements in cochlear biomechanics. *Journal of Acoustical Society of America*, Vol. 79, No. 5, pp. 1472–1480, DOI: 10.1121/1.393674.
  18. Elliot, S.J., Ku, E.M., & Lineton, B.A. (2007). A state space model for cochlear mechanics. *Journal of Acoustical Society of America*, Vol. 122, No. 5, pp. 2759–2771. DOI: 10.1121/1.2783125.
  19. Elliott, S.J., Lineton, B., Ni, G. (2011). Fluid coupling in a discrete model of cochlear mechanics. *Journal of Acoustical Society of America*, Vol. 130, No. 3, pp. 1441–1451. DOI: 10.1121/1.3607420.
  20. Ku, E.M., Elliot, S.J. & Lineton, B.A. (2008). Statistics of instabilities in a state space model of the human cochlea. *Journal of Acoustical Society of America*, Vol. 124, No. 2, pp. 1068–1079. DOI 10.1121/1.2939133.
  21. Haque, S. & Togneri, R. (2010). A feature extraction method for automatic speech recognition based on the cochlear nucleus, *Annual Conference of the International Speech Communication Association*.
  22. Harczos, T., Szepannek, G., & Klefenz, F. (2007). Towards Automatic Speech Recognition based on Cochlear Traveling Wave Delay Trajectories. Auditory signal processing in hearing-impaired listeners, *1st International Symposium on Auditory and Audiological Research*, Vol. 1, pp. 83–93.
  23. Keener, J. & Sneyd, J. (2009). *Mathematical Physiology*. Springer. DOI: 10.1007/978-0-387-79388-7.
  24. von-Békésy, G. (1960). *Experiments in hearing*. Mc Graw Hill (USA)
  25. Jiménez-Hernández, M., Oropeza-Rodríguez, J.L., Suárez-Guerra, S., & Barrón-Fernández, R. (2012). Computational Model of the Cochlea using Resonance Analysis. *Journal Revista Mexicana Ingeniería Biomédica*, Vol. 33, Num. 2, pp. 77–86.
  26. Greenwood, D.D. (1990). A cochlear frequency-position function for several species—29 years later. *J. Acoust. Soc. Am.*, Vol. 87, No. 6, pp. 2592–2605. DOI: 10.1121/1.399052.
  27. Greenwood, D.D. (1961). Critical bandwidth and the frequency coordinates of the basilar membrane. *J. Acoust. Soc. Am.*, Vol. 33, No. 10, pp. 1344–1356. DOI: 10.1121/1.1908437.
  28. Greenwood, D.D. (1996). Comparing octaves, frequency ranges, and cochlear-map curvature across species. *Hear. Res.*, Vol. 94, No. 1-2, pp. 157–162. DOI: 10.1016/0378-5955(95)00229-4.
  29. Greenwood, D.D. (1997). The Mel Scale's disqualifying bias and a consistency of pitch-difference equisections in 1956 with equal cochlear distances and equal frequency ratios. *Hear. Res.*, Vol. 103, No. 1-2, pp. 199–224. DOI: 10.1016/S0378-5955(96)00175-X.
  30. Kinsler, E.L., Frey, R.A., Coppens, B.A., & Sanders, V.J. (2000). *Fundamentals of Acoustics*. 4th edition. John Wiley & Sons Inc., pp. 312–315.
  31. Oropeza-Rodríguez, J.L. & Reyes-Saldana, J.F. (2013). Using a Model of the Cochlea Based in the Micro and Macro Mechanical to Find Parameters for Automatic Speech Recognition. *MICAI (Special Sessions)*, pp. 171–177. DOI: 10.1109/MICAI.2013.39.
  32. Yost, W.A. (2006) *Fundamentals of hearing: An introduction*. Fifth Edition, Academic Press (USA).

ISSN 2007-9737

1114 *José Luis Oropeza Rodríguez, Sergio Suárez Guerra*

*Article received on 15/06/2018; accepted on 21/12/2018.  
Corresponding author is José Luis Oropeza Rodríguez.*

## Modelado para traves de sección transversal rectangular con cartelas parabólicas: Parte 2

Ricardo Sandoval Rivas, Arnulfo Luévanos Rojas, Sandra López Chavarría,  
Manuel Medina Elizondo

Universidad Autónoma de Coahuila,  
Instituto de Investigaciones Multidisciplinarias, Torreón, Coahuila,  
México

{ricardo\_sandoval\_rivas, arnulfol\_2007, sandylopez5}@hotmail.com,  
drmanuelmediana@yahoo.com.mx

**Resumen.** La parte 1 de este documento se presentó un modelo matemático para traves de sección transversal rectangular con cartelas parabólicas (simétricas o no simétricas) sujetas a una carga uniformemente distribuida tomando en cuenta las deformaciones por flexión y cortante para obtener los momentos de empotramiento, factores de transporte y factores de rigidez. En este trabajo se presenta un modelo matemático para el mismo tipo de traves de sección transversal con cartelas parabólicas bajo una carga concentrada localizada en cualquier parte de la viga tomando en cuenta las deformaciones por flexión y cortante para obtener los momentos de empotramiento usando el mismo procedimiento. El modelo tradicional considera las deformaciones por flexión solamente. También, una comparación se realiza entre el modelo tradicional y el modelo propuesto. Además de la eficacia y la precisión del modelo desarrollado, una ventaja significativa es que los momentos de empotramiento se calculan para cualquier sección transversal rectangular de la viga usando la ecuación matemática presentada en este documento, que es la parte principal de esta investigación.

**Palabras clave.** Miembros rectangulares, carga concentrada, cartelas parabólicas, deformaciones por flexión y cortante, momentos de empotramiento.

### Modeling for Beams of Rectangular Cross Section with Parabolic Haunches: Part 2

**Abstract.** The part 1 of this paper is presented a mathematical model for beams of rectangular cross section with parabolic haunches (symmetric or non-symmetric) subjected to a uniformly distributed load

taking into account the bending and shear deformations to obtain the fixed-end moments, carry-over factors and stiffness factors. In this paper is presented a mathematical model for the same type of beams of cross section with parabolic haunches under a concentrated load located anywhere on the beam taking into account the bending and shear deformations to obtain the fixed-end moments using the same procedure. The traditional model considers only bending deformations, and others authors present tables considering the bending and shear deformations, but are restricted to certain relationships. Also, a comparison is made between the traditional model and the proposed model. Besides the effectiveness and accuracy of the developed models, a significant advantage is that fixed-end moments are calculated for any rectangular cross section of the beam using the mathematical equation presented in this paper, which is the main part of this research.

**Keywords.** Rectangular members, concentrated load, parabolic haunches, bending and shear deformations, fixed-end moments.

### 1. Introducción

Una de las principales preocupaciones de la ingeniería estructural es proponer métodos elásticos fiables para modelar satisfactoriamente a los miembros de sección transversal variable, de tal manera que se tenga la certeza en la determinación de los elementos mecánicos, tales como: deformaciones y desplazamientos que permitan diseñar adecuadamente este tipo de miembros.

Las traves acarteladas de concreto reforzado ofrecen las siguientes ventajas con respecto a las

vigas prismáticas: a) Aumentan sustancialmente la rigidez lateral, b) Promover un uso más eficiente del concreto y del acero de refuerzo longitudinal, c) Reducir el peso del edificio, optimizar la resistencia y la estabilidad o para cumplir los requisitos arquitectónicos y funcionales específicos, d) Facilitar la colocación de instalaciones eléctricas, de aire acondicionado y sanitarias en el edificio (lo cual reduce la altura del entrepiso).

Los miembros de sección transversal rectangular sometida a una carga concentrada su principal aplicación se encuentra en las cargas vivas (cargas móviles) de los puentes correspondientes a las cargas concentradas transmitidos por los vehículos a través de sus ruedas hacia la superficie de la carretera en el tablero.

Durante el siglo pasado, entre 1950 y 1960 se desarrollaron varias ayudas de diseño, como las presentadas por Guldán [1], y las tablas más populares publicados por Portland Cement Association (PCA) en 1958 "Handbook" [2].

Los documentos más relevantes que tratan el tema de los elementos estructurales con sección transversal variable se muestran en la parte 1 [3-18].

Los métodos tradicionales toman en cuenta las deformaciones por flexión y las deformaciones por cortante se desprecian [19-21].

Este trabajo presenta un modelo matemático para traveses de sección transversal rectangular con cartelas parabólicas (simétricas o no simétricas) bajo una carga concentrada localizada en cualquier parte de la viga tomando en cuenta las deformaciones por flexión y cortante para obtener los momentos de empotramiento, que es la novedad de esta investigación. Las propiedades de la sección transversal rectangular de la viga varían a lo largo de su eje "x", es decir, el ancho "b" es constante y la altura "h" varía a lo largo de la viga en tres partes diferentes, las partes extremas tienen una variación parabólica, y la parte central es constante. Las ecuaciones de compatibilidad y equilibrio se utilizan para resolver este tipo de problemas, y las deformaciones en cualquier parte de la viga se encuentran por medio del principio del trabajo virtual mediante la integración exacta para obtener los momentos de empotramiento.

Algunos resultados se obtienen usando el software "Derive". El modelo tradicional considera las deformaciones por flexión solamente. También, una comparación se realiza entre el modelo tradicional y el modelo propuesto para observar las diferencias. Además de la eficacia y la precisión del modelo desarrollado, una ventaja significativa es que los momentos de empotramiento se calculan para cualquier sección transversal rectangular de la viga usando la ecuación matemática presentada en este documento, que es la parte principal de esta investigación.

## 2. Modelo propuesto

### 2.1. Momentos de empotramiento para una carga concentrada

La Figura 1 de la parte 1 muestra una viga en elevación y también presenta su sección transversal rectangular tomando en cuenta el ancho "b" es constante y la altura " $h_x$ " varía a lo largo de su eje "x" de forma parabólica en tres partes diferentes.

En la Figura 1(a) para los casos 1, 2 y 3 (Caso 1: cuando la carga  $P$  se localiza de  $0 \leq x \leq a$ , Caso 2: cuando la carga  $P$  se encuentra de  $a \leq x \leq L - c$ , Caso 3: cuando la carga  $P$  se ubica de  $L - c \leq x \leq L$ ) se muestra la viga "AB" sometida a una carga concentrada localizada en cualquier parte de la viga y empotrada en sus extremos. Los momentos de empotramiento en sus extremos se encuentran mediante la suma de los efectos.

Los momentos se consideran positivos, cuando giran en sentido inverso y negativo cuando giran en sentido horario. En la Figura 1(b) para los tres casos se presenta la misma viga simplemente apoyada en sus extremos bajo la carga aplicada para obtener las rotaciones " $\theta_{A1}$ " y " $\theta_{B1}$ " para el caso 1, " $\theta_{A2}$ " y " $\theta_{B2}$ " para el caso 2, " $\theta_{A3}$ " y " $\theta_{B3}$ " para el caso 3.

Ahora, las rotaciones " $f_{11}$ " y " $f_{21}$ " son causados por un momento unitario aplicado en el soporte "A", según las Figura 1(c), y en cuanto a " $f_{12}$ " y " $f_{22}$ " son causados por el momento unitario aplicado en el soporte "B", esto se observa en las Figura 1(d) [10-18].

Las ecuaciones de compatibilidad y equilibrio para la viga son [22-24]:

para  $0 \leq x \leq a$ :

$$-f_{11}M_{AB} + f_{12}M_{BA} = \theta_{A1}, \quad (1)$$

$$-f_{21}M_{AB} + f_{22}M_{BA} = \theta_{B1}, \quad (2)$$

para  $a \leq x \leq L - c$ :

$$-f_{11}M_{AB} + f_{12}M_{BA} = \theta_{A2}, \quad (3)$$

$$-f_{21}M_{AB} + f_{22}M_{BA} = \theta_{B2}, \quad (4)$$

para  $L - c \leq x \leq L$ :

$$-f_{11}M_{AB} + f_{12}M_{BA} = \theta_{A3}, \quad (5)$$

$$-f_{21}M_{AB} + f_{22}M_{BA} = \theta_{B3}. \quad (6)$$

Las vigas de la Figura 1(b) son analizadas para encontrar " $\theta_{Aj}$ " y " $\theta_{Bj}$ ", donde  $j$  toma los valores 1, 2 y 3, el principio del trabajo virtual y tomando en cuenta las deformaciones de flexión y cortante se utiliza para obtener las rotaciones.

Ahora los valores de " $\theta_{Aj}$ " y " $\theta_{Bj}$ " para los miembros no prismáticos se encuentran por las siguientes ecuaciones [7]:

$$\theta_A = \int_0^L \frac{V_x V_1}{GA_{sx}(x)} dx + \int_0^L \frac{M_x M_1}{EI_z(x)} dx, \quad (7)$$

$$\theta_B = \int_0^L \frac{V_x V_2}{GA_{sx}(x)} dx + \int_0^L \frac{M_x M_2}{EI_z(x)} dx, \quad (8)$$

donde  $E$  es el módulo de elasticidad,  $G$  es el módulo de cortante,  $V_x$  y  $M_x$  son la fuerza cortante y el momento de la carga real,  $V_1$  y  $M_1$  son la fuerza cortante y el momento debido al momento unitario aplicado en el apoyo "A",  $V_2$  y  $M_2$  son la fuerza cortante y el momento debido al momento unitario aplicado en el apoyo "B" a una distancia " $x$ ".

Tabla 1 presenta las ecuaciones de las fuerzas cortantes y los momentos en cualquier parte de la viga sobre el eje " $x$ " son [25].

Sustituyendo los valores de las Tablas 1 (parte 1 y parte 2) en las ecuaciones (7, 8) para obtener las rotaciones de " $\theta_{A1}$ ", " $\theta_{B1}$ ", " $\theta_{A2}$ ", " $\theta_{B2}$ ", " $\theta_{A3}$ " y " $\theta_{B3}$ ", estas se muestran de la siguiente manera:

$$\begin{aligned} \theta_{A1} = & \frac{6P}{5bGL^2} \left\{ \frac{a(L-e)}{h^{1/2}u^{1/2}} \operatorname{Atan} \left( \frac{u}{h} \right)^{1/2} - \right. \\ & \frac{aL}{h^{1/2}u^{1/2}} \operatorname{Atan} \left[ \frac{(a-e)u^{1/2}}{ah^{1/2}} \right] - \frac{ce}{h^{1/2}s^{1/2}} \operatorname{Atan} \left( \frac{s}{h} \right)^{1/2} - \\ & \left. \frac{e(L-a-c)}{h} \right\} - \frac{12P}{bEL^2} \left\{ \frac{a^2(L-e)[3u(L-a)-ah]}{8h^2u^2} \operatorname{Atan} \left( \frac{u}{h} \right)^{1/2} + \right. \\ & \left. \frac{c^3e(h+3s)}{8h^2s^2} \operatorname{Atan} \left( \frac{s}{h} \right)^{1/2} + \right. \\ & \left. \frac{aL[a^2h-3u(a-e)(L-a)]}{8h^2u^2} \operatorname{Atan} \left[ \frac{(a-e)u^{1/2}}{ah^{1/2}} \right] - \right. \\ & \left. \frac{a^2L(L-e)(a-e)^2}{8h^2[a^2h+u(a-e)^2]} + \frac{a^2e(L-a)}{8h^2u} - \frac{c^3e}{8h^2s} + \frac{a^2L(L-e)}{8h^2(h+u)} + \right. \\ & \left. \left. \frac{e[(L-a)^3-c^3]}{3h^3} \right\}, \end{aligned} \quad (9)$$

$$\begin{aligned} \theta_{B1} = & \frac{6P}{5bGL^2} \left\{ \frac{a(L-e)}{h^{1/2}u^{1/2}} \operatorname{Atan} \left( \frac{u}{h} \right)^{1/2} - \right. \\ & \frac{aL}{h^{1/2}u^{1/2}} \operatorname{Atan} \left[ \frac{(a-e)u^{1/2}}{ah^{1/2}} \right] - \frac{ce}{h^{1/2}s^{1/2}} \operatorname{Atan} \left( \frac{s}{h} \right)^{1/2} - \\ & \left. \frac{e(L-a-c)}{h} \right\} + \frac{12P}{bEL^2} \left\{ \frac{a^3(L-e)(h+3u)}{8h^{5/2}u^{3/2}} \operatorname{Atan} \left( \frac{u}{h} \right)^{1/2} + \right. \\ & \left. \frac{c^2e[3s(L-c)-ch]}{8h^{5/2}s^{3/2}} \operatorname{Atan} \left( \frac{s}{h} \right)^{1/2} - \right. \\ & \left. \frac{a^2L[3u(a-e)+ah]}{8h^{5/2}u^{3/2}} \operatorname{Atan} \left[ \frac{(a-e)u^{1/2}}{ah^{1/2}} \right] - \right. \\ & \left. \frac{a^2eL(a-e)^2}{8h^2[a^2h+u(a-e)^2]} - \frac{a^2e(L-a)}{8h^2u} + \frac{c^3e}{8h^2s} + \frac{c^2eL}{8h^2(h+s)} + \right. \\ & \left. \left. \frac{e(2a^3-3a^2L+2c^3-3c^2L+L^3)}{6h^3} \right\}, \end{aligned} \quad (10)$$

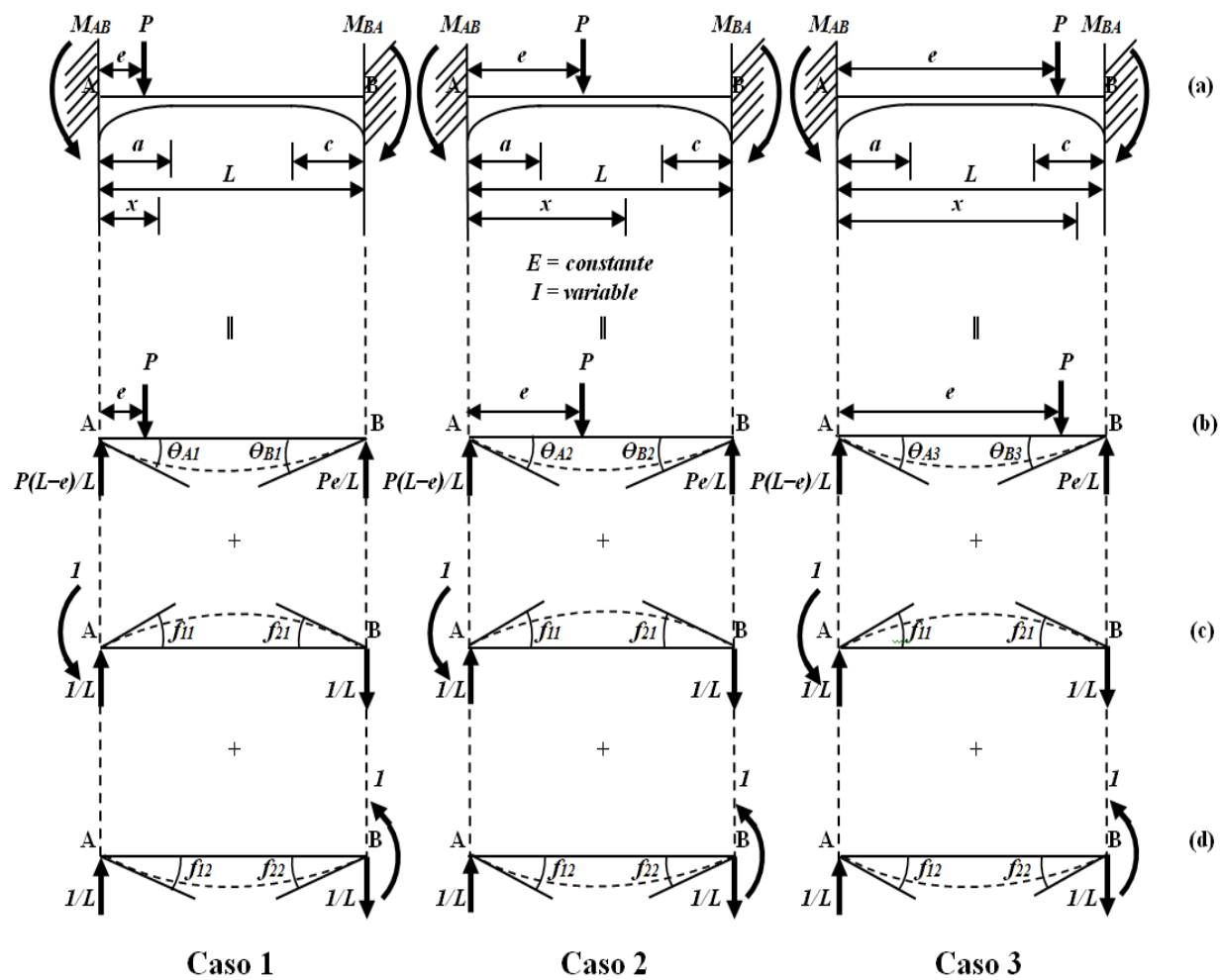
$$\begin{aligned} \theta_{A2} = & \frac{6P}{5bGL^2} \left\{ \frac{a(L-e)}{h^{1/2}u^{1/2}} \operatorname{Atan} \left( \frac{u}{h} \right)^{1/2} - \right. \\ & \frac{ce}{h^{1/2}s^{1/2}} \operatorname{Atan} \left( \frac{s}{h} \right)^{1/2} - \frac{a(L-e)-ce}{h} \left\} - \right. \\ & \left. \frac{12P}{bEL^2} \left\{ \frac{a^2(L-e)[3u(L-a)-ah]}{8h^{5/2}u^{3/2}} \operatorname{Atan} \left( \frac{u}{h} \right)^{1/2} + \right. \right. \\ & \left. \frac{c^3e(h+3s)}{8h^{5/2}s^{3/2}} \operatorname{Atan} \left( \frac{s}{h} \right)^{1/2} + \frac{a^3(L-e)}{8h^2u} + \frac{a^2L(L-e)}{8h^2(h+u)} - \right. \\ & \left. \left. \frac{c^3e}{8h^2s} + \frac{(L-e)(2a^3-3a^2L-2e^3+3e^2L)}{6h^3} + \frac{e[(L-e)^3-c^3]}{3h^3} \right\}, \end{aligned} \quad (11)$$

$$\begin{aligned} \theta_{B2} = & \frac{6P}{5bGL^2} \left\{ \frac{a(L-e)}{h^{1/2}u^{1/2}} \operatorname{Atan} \left( \frac{u}{h} \right)^{1/2} - \right. \\ & \frac{ce}{h^{1/2}s^{1/2}} \operatorname{Atan} \left( \frac{s}{h} \right)^{1/2} - \frac{a(L-e)-ce}{h} \left\} + \right. \\ & \left. \frac{12P}{bEL^2} \left\{ \frac{a^3(L-e)(h+3u)}{8h^{5/2}u^{3/2}} \operatorname{Atan} \left( \frac{u}{h} \right)^{1/2} + \right. \right. \\ & \left. \frac{c^2e[3s(L-c)-ch]}{8h^{5/2}s^{3/2}} \operatorname{Atan} \left( \frac{s}{h} \right)^{1/2} - \frac{a^3(L-e)}{8h^2u} + \frac{c^2eL}{8h^2(h+s)} + \right. \\ & \left. \left. \frac{c^3e}{8h^2s} + \frac{e(2c^3-3c^2L+2e^3-3e^2L+L^3)}{6h^3} - \frac{(a^3-e^3)(L-e)}{3h^3} \right\}, \end{aligned} \quad (12)$$

Los coeficientes de flexibilidades se muestran en las ecuaciones (10, 11, 12) de la parte 1.

Tabla 1. Fuerzas cortantes y momentos

Concepto	Ecuaciones			
Fuerza cortante	A la izquierda de P	$V_x = \frac{P(L-e)}{L}$	$V_1 = \frac{1}{L}$	$V_2 = \frac{1}{L}$
	A la derecha de P	$V_x = -\frac{Pe}{L}$		
Momento	A la izquierda de P	$M_x = \frac{P(L-e)x}{L}$	$M_1 = -\frac{(L-x)}{L}$	$M_2 = \frac{x}{L}$
	A la derecha de P	$M_x = \frac{Pe(L-x)}{L}$		



**Fig. 1.** Viga doblemente empotrada bajo una carga concentrada ubicada en cualquier parte de la viga: Caso 1 es cuando la carga  $P$  se localiza de  $0 \leq x \leq a$ . Caso 2 es cuando la carga  $P$  se encuentra de  $a \leq x \leq L - c$ . Caso 3 es cuando la carga  $P$  se ubica de  $L - c \leq x \leq L$



$$\begin{aligned} \theta_{A3} &= \frac{6P}{5bGL^2} \left\{ \frac{a(L-e)}{h^{1/2}u^{1/2}} \operatorname{Atan}\left(\frac{u}{h}\right)^{1/2} \right. \\ &+ \frac{cL}{h^{1/2}s^{1/2}} \operatorname{Atan}\left[\frac{(c+e-L)s^{1/2}}{ch^{1/2}}\right] \\ &- \frac{ce}{h^{1/2}s^{1/2}} \operatorname{Atan}\left(\frac{s}{h}\right)^{1/2} + \frac{(L-e)(L-a-c)}{h} \left. \right\} \\ &- \frac{12P}{bEL^2} \left\{ \frac{c^3e(h+3s)}{8h^{5/2}s^{3/2}} \operatorname{Atan}\left(\frac{s}{h}\right)^{1/2} \right. \\ &- \frac{a^2(L-e)[3u(a-L)+ah]}{8h^{5/2}u^{3/2}} \operatorname{Atan}\left(\frac{u}{h}\right)^{1/2} \\ &- \frac{c^2L[3s(c+e-L)+ch]}{8h^{5/2}s^{3/2}} \operatorname{Atan}\left[\frac{(c+e-L)s^{1/2}}{ch^{1/2}}\right] \\ &+ \frac{a^3(L-e)}{8h^2u} + \frac{a^2L(L-e)}{8h^2(h+u)} - \frac{c^2(L-e)(L-c)}{8h^2s} \\ &+ \frac{(L-e)(2a^3-3a^2L+2c^3-3c^2L+L^3)}{6h^3} \\ &\left. - \frac{c^2L(L-e)(c+e-L)^2}{8h^2[s(c+e-L)^2+c^2h]} \right\}, \end{aligned} \quad (13)$$

$$\begin{aligned} \theta_{B3} &= \frac{6P}{5bGL^2} \left\{ \frac{a(L-e)}{h^{1/2}u^{1/2}} \operatorname{Atan}\left(\frac{u}{h}\right)^{1/2} \right. \\ &+ \frac{cL}{h^{1/2}s^{1/2}} \operatorname{Atan}\left[\frac{(c+e-L)s^{1/2}}{ch^{1/2}}\right] \\ &- \frac{ce}{h^{1/2}s^{1/2}} \operatorname{Atan}\left(\frac{s}{h}\right)^{1/2} + \frac{(L-e)(L-a-c)}{h} \left. \right\} \\ &+ \frac{12P}{bEL^2} \left\{ \frac{a^3(L-e)(h+3u)}{8h^{5/2}u^{3/2}} \operatorname{Atan}\left(\frac{u}{h}\right)^{1/2} \right. \\ &+ \frac{c^2e[3s(L-c)-ch]}{8h^{5/2}s^{3/2}} \operatorname{Atan}\left(\frac{s}{h}\right)^{1/2} \\ &- \frac{cL[3s(L-c)(c+e-L)-c^2h]}{8h^{5/2}s^{3/2}} \operatorname{Atan}\left[\frac{(c+e-L)s^{1/2}}{ch^{1/2}}\right] \\ &- \frac{a^3(L-e)}{8h^2u} + \frac{c^2eL}{8h^2(h+s)} + \frac{c^2(L-e)(L-c)}{8h^2s} \\ &\left. - \frac{(L-e)[a^3-(L-c)^3]}{3h^3} - \frac{c^2eL(c+e-L)^2}{8h^2[s(c+e-L)^2+c^2h]} \right\}. \end{aligned} \quad (14)$$

### Caso 1: cuando la carga $P$ se localiza de $0 \leq x \leq a$

La ecuación (9) y las ecuaciones (10, 12) de la parte 1 correspondientes al apoyo "A" se sustituyen en la ecuación (1), y la ecuación (10) y las ecuaciones (11, 12) de la parte 1 correspondientes al apoyo "B" se sustituyen en la ecuación (2). Subsecuentemente, las ecuaciones generadas se resuelven para obtener los valores de " $M_{AB}$ " y " $M_{BA}$ ". Estas se presentan como sigue:

$$M_{AB} = \frac{f_{22}\theta_{A1} - f_{12}\theta_{B1}}{(f_{12})^2 - f_{11}f_{22}}, \quad (15)$$

$$M_{BA} = \frac{f_{12}\theta_{A1} - f_{11}\theta_{B1}}{(f_{12})^2 - f_{11}f_{22}}. \quad (16)$$

### Caso 2: cuando la carga $P$ se encuentra de $a \leq x \leq L - c$

La ecuación (11) y las ecuaciones (10, 12) de la parte 1 correspondientes al apoyo "A" se sustituyen en la ecuación (3), y la ecuación (12) y las ecuaciones (11, 12) de la parte 1 correspondientes al apoyo "B" se sustituyen en la ecuación (4). Subsecuentemente, las ecuaciones generadas se resuelven para obtener los valores de " $M_{AB}$ " y " $M_{BA}$ ". Estas se muestran de la siguiente manera:

$$M_{AB} = \frac{f_{22}\theta_{A2} - f_{12}\theta_{B2}}{(f_{12})^2 - f_{11}f_{22}}, \quad (17)$$

$$M_{BA} = \frac{f_{12}\theta_{A2} - f_{11}\theta_{B2}}{(f_{12})^2 - f_{11}f_{22}}. \quad (18)$$

### Caso 3: cuando la carga $P$ se ubica de $L - c \leq x \leq L$

La ecuación (13) y las ecuaciones (10, 12) de la parte 1 correspondientes al apoyo "A" se sustituyen en la ecuación (5), y la ecuación (14) y las ecuaciones (11, 12) de la parte 1 correspondientes al apoyo "B" se sustituyen en la ecuación (6). Subsecuentemente, las ecuaciones generadas se resuelven para obtener los valores de " $M_{AB}$ " y " $M_{BA}$ ". Estas se presentan como sigue:

$$M_{AB} = \frac{f_{22}\theta_{A3} - f_{12}\theta_{B3}}{(f_{12})^2 - f_{11}f_{22}}, \quad (19)$$

$$M_{BA} = \frac{f_{12}\theta_{A3} - f_{11}\theta_{B3}}{(f_{12})^2 - f_{11}f_{22}}. \quad (20)$$

## 3. Validación del modelo propuesto

En las Tablas 2 y 3 se muestran los resultados de los dos modelos para los factores de los mo-

Tabla 2. Momentos de empotramiento para  $h = 0.10L$ 

M <sub>AB</sub> = m <sub>AB</sub> PL; M <sub>BA</sub> = m <sub>BA</sub> PL																					
c	s/ h	e = 0.1L				e = 0.3L				e = 0.5L				e = 0.7L				e = 0.9L			
		m <sub>AB</sub>		m <sub>BA</sub>		m <sub>AB</sub>		m <sub>BA</sub>		m <sub>AB</sub>		m <sub>BA</sub>		m <sub>AB</sub>		m <sub>BA</sub>		m <sub>AB</sub>		m <sub>BA</sub>	
		M P	M T	M P	M T	M P	M T	M P	M T	M P	M T	M P	M T	M P	M T	M P	M T	M P	M T	M P	M T
a = 0.2L; u/h = 1.0																					
0.2 L	0.4	0.09 26	0.09 38	0.00 44	0.00 33	0.18 68	0.18 91	0.05 24	0.05 02	0.15 68	0.15 72	0.12 65	0.12 61	0.07 33	0.07 15	0.16 01	0.16 18	0.00 87	0.00 73	0.08 65	0.08 77
	0.6	0.09 24	0.09 37	0.00 47	0.00 35	0.18 49	0.18 72	0.05 57	0.05 35	0.15 24	0.15 27	0.13 42	0.13 39	0.06 83	0.06 63	0.16 89	0.17 08	0.00 72	0.00 58	0.08 89	0.09 02
	1.0	0.09 22	0.09 35	0.00 51	0.00 38	0.18 20	0.18 44	0.06 08	0.05 84	0.14 59	0.14 59	0.14 59	0.14 59	0.06 08	0.05 84	0.18 20	0.18 44	0.00 51	0.00 38	0.09 22	0.09 35
	1.5	0.09 20	0.09 33	0.00 55	0.00 41	0.17 95	0.18 19	0.06 53	0.06 28	0.14 02	0.13 99	0.15 61	0.15 63	0.05 44	0.05 18	0.19 33	0.19 60	0.00 37	0.00 25	0.09 45	0.09 58
	2.0	0.09 18	0.09 32	0.00 58	0.00 44	0.17 78	0.18 01	0.06 85	0.06 60	0.13 61	0.13 58	0.16 35	0.16 38	0.05 00	0.04 72	0.20 13	0.20 42	0.00 28	0.00 17	0.09 59	0.09 71
0.3 L	0.4	0.09 24	0.09 37	0.00 48	0.00 35	0.18 49	0.18 73	0.05 60	0.05 37	0.15 29	0.15 32	0.13 42	0.13 39	0.06 97	0.06 78	0.16 67	0.16 86	0.00 86	0.00 73	0.08 64	0.08 77
	0.6	0.09 22	0.09 35	0.00 52	0.00 39	0.18 22	0.18 45	0.06 11	0.05 87	0.14 67	0.14 67	0.14 56	0.14 55	0.06 31	0.06 09	0.17 86	0.18 08	0.00 71	0.00 57	0.08 89	0.09 02
	1.0	0.09 18	0.09 32	0.00 60	0.00 44	0.17 78	0.18 01	0.06 94	0.06 69	0.13 69	0.13 65	0.16 39	0.16 43	0.05 29	0.05 02	0.19 71	0.20 00	0.00 50	0.00 37	0.09 21	0.09 36
	1.5	0.09 14	0.09 29	0.00 67	0.00 50	0.17 37	0.17 60	0.07 72	0.07 46	0.12 79	0.12 72	0.18 10	0.18 19	0.04 40	0.04 10	0.21 36	0.21 70	0.00 35	0.00 23	0.09 45	0.09 59
	2.0	0.09 12	0.09 27	0.00 72	0.00 54	0.17 07	0.17 30	0.08 32	0.08 05	0.12 13	0.12 03	0.19 38	0.19 51	0.03 77	0.03 45	0.22 54	0.22 93	0.00 27	0.00 16	0.09 59	0.09 72
a = 0.5L; u/h = 1.0																					
0.2 L	0.4	0.09 13	0.09 28	0.00 46	0.00 34	0.20 96	0.21 27	0.04 01	0.03 78	0.20 10	0.20 23	0.10 72	0.10 62	0.10 00	0.09 79	0.14 92	0.15 07	0.01 24	0.01 04	0.08 50	0.08 65
	0.6	0.09 10	0.09 26	0.00 49	0.00 37	0.20 76	0.21 08	0.04 28	0.04 04	0.19 58	0.19 70	0.11 43	0.11 34	0.09 35	0.09 10	0.15 81	0.15 99	0.01 03	0.00 83	0.08 76	0.08 92
	1.0	0.09 07	0.09 24	0.00 54	0.00 41	0.20 45	0.20 77	0.04 71	0.04 46	0.18 80	0.18 88	0.12 52	0.12 46	0.08 36	0.08 06	0.17 15	0.17 39	0.00 75	0.00 55	0.09 12	0.09 27
	1.5	0.09 04	0.09 21	0.00 58	0.00 44	0.20 18	0.20 50	0.05 10	0.04 83	0.18 11	0.18 16	0.13 49	0.13 45	0.07 52	0.07 18	0.18 32	0.18 61	0.00 54	0.00 36	0.09 37	0.09 53
	2.0	0.09 01	0.09 19	0.00 62	0.00 47	0.19 99	0.20 31	0.05 37	0.05 10	0.17 62	0.17 65	0.14 20	0.14 17	0.06 93	0.06 56	0.19 16	0.19 47	0.00 42	0.00 24	0.09 53	0.09 68
0.5 L	0.4	0.09 08	0.09 24	0.00 55	0.00 41	0.20 54	0.20 87	0.04 88	0.04 42	0.19 12	0.19 25	0.12 16	0.12 05	0.09 17	0.08 96	0.15 80	0.15 98	0.01 26	0.01 06	0.08 37	0.08 53
	0.6	0.09 02	0.09 20	0.00 63	0.00 47	0.20 12	0.20 45	0.05 35	0.05 06	0.18 11	0.18 20	0.13 67	0.13 60	0.08 16	0.07 91	0.17 16	0.17 39	0.01 06	0.00 86	0.08 60	0.08 78
	1.0	0.08 93	0.09 13	0.00 79	0.00 59	0.19 37	0.19 70	0.06 59	0.06 26	0.16 39	0.16 39	0.16 39	0.16 39	0.06 59	0.06 26	0.19 37	0.19 70	0.00 79	0.00 59	0.08 93	0.09 13
	1.5	0.08 83	0.09 05	0.00 96	0.00 72	0.18 60	0.18 91	0.07 94	0.07 59	0.14 66	0.14 56	0.19 27	0.19 39	0.05 20	0.04 79	0.21 43	0.21 87	0.00 58	0.00 39	0.09 19	0.09 40
	2.0	0.08 75	0.08 99	0.01 12	0.00 84	0.17 95	0.18 25	0.09 12	0.08 77	0.13 29	0.13 07	0.21 68	0.21 93	0.04 21	0.03 76	0.22 94	0.23 48	0.00 46	0.00 27	0.09 35	0.09 57

Tabla 3. Momentos de empotramiento para  $h = 0.20L$ 

M <sub>AB</sub> = m <sub>AB</sub> PL; M <sub>BA</sub> = m <sub>BA</sub> PL																					
c	s/ h	e = 0.1L				e = 0.3L				e = 0.5L				e = 0.7L				e = 0.9L			
		m <sub>AB</sub>		m <sub>BA</sub>		m <sub>AB</sub>		m <sub>BA</sub>		m <sub>AB</sub>		m <sub>BA</sub>		m <sub>AB</sub>		m <sub>BA</sub>		m <sub>AB</sub>		m <sub>BA</sub>	
		MP	MT	MP	MT	MP	MT	MP	MT	MP	MT	MP	MT	MP	MT	MP	MT	MP	MT	MP	MT
a = 0.2L; u/h = 1.0																					
0.2L	0.4	0.08	0.09	0.00	0.00	0.18	0.18	0.05	0.05	0.15	0.15	0.12	0.12	0.07	0.07	0.15	0.16	0.01	0.00	0.08	0.08
		95	38	73	33	08	91	79	02	58	72	75	61	79	15	58	18	23	73	32	77
	0.6	0.08	0.09	0.00	0.00	0.17	0.18	0.06	0.05	0.15	0.15	0.13	0.13	0.07	0.06	0.16	0.17	0.01	0.00	0.08	0.09
		92	37	78	35	89	72	14	35	18	27	48	39	34	63	39	08	07	58	56	02
	1.0	0.08	0.09	0.00	0.00	0.17	0.18	0.06	0.05	0.14	0.14	0.14	0.14	0.06	0.05	0.17	0.18	0.00	0.00	0.08	0.09
		89	35	85	38	61	44	67	84	59	59	59	59	67	84	61	44	85	38	89	35
	1.5	0.08	0.09	0.00	0.00	0.17	0.18	0.07	0.06	0.14	0.13	0.15	0.15	0.06	0.05	0.18	0.19	0.00	0.00	0.09	0.09
		86	33	91	41	37	19	14	28	07	99	55	63	09	18	66	60	67	25	14	58
	2.0	0.08	0.09	0.00	0.00	0.17	0.18	0.07	0.06	0.13	0.13	0.16	0.16	0.05	0.04	0.19	0.20	0.00	0.00	0.09	0.09
		83	32	95	44	19	01	47	60	70	58	25	38	69	72	39	42	56	17	29	71
0.3L	0.4	0.08	0.09	0.00	0.00	0.17	0.18	0.06	0.05	0.15	0.15	0.13	0.13	0.07	0.06	0.16	0.16	0.01	0.00	0.08	0.08
		92	37	79	35	89	73	18	37	22	32	49	39	45	78	21	86	21	73	30	77
	0.6	0.08	0.09	0.00	0.00	0.17	0.18	0.06	0.05	0.14	0.14	0.14	0.14	0.06	0.06	0.17	0.18	0.01	0.00	0.08	0.09
		89	35	86	39	62	45	71	87	66	67	57	55	86	09	30	08	05	57	54	02
	1.0	0.08	0.09	0.00	0.00	0.17	0.18	0.07	0.06	0.13	0.13	0.16	0.16	0.05	0.05	0.19	0.20	0.00	0.00	0.08	0.09
		83	32	98	44	19	01	57	69	78	65	29	43	94	02	00	00	83	37	86	36
	1.5	0.08	0.09	0.01	0.00	0.16	0.17	0.08	0.07	0.12	0.12	0.17	0.18	0.05	0.04	0.20	0.21	0.00	0.00	0.09	0.09
		78	29	09	50	80	60	38	46	98	72	88	19	14	10	51	70	65	23	11	59
	2.0	0.08	0.09	0.01	0.00	0.16	0.17	0.08	0.08	0.12	0.12	0.19	0.19	0.04	0.03	0.21	0.22	0.00	0.00	0.09	0.09
		74	27	17	54	51	30	98	05	40	03	07	51	57	45	60	93	54	16	27	72
a = 0.5L; u/h = 1.0																					
0.5L	0.4	0.08	0.09	0.00	0.00	0.20	0.21	0.04	0.03	0.19	0.20	0.10	0.10	0.09	0.14	0.15	0.01	0.01	0.08	0.08	
		75	28	74	34	19	27	58	78	76	23	97	62	52	79	53	07	74	04	13	65
	0.6	0.08	0.09	0.00	0.00	0.19	0.21	0.04	0.04	0.19	0.19	0.11	0.11	0.09	0.09	0.15	0.15	0.01	0.00	0.08	0.08
		71	26	79	37	99	08	87	04	30	70	65	34	95	10	34	99	53	83	38	92
	1.0	0.08	0.09	0.00	0.00	0.19	0.20	0.05	0.04	0.18	0.18	0.12	0.12	0.09	0.08	0.16	0.17	0.01	0.00	0.08	0.09
		66	24	86	41	69	77	33	46	60	88	68	46	09	06	57	39	22	55	74	27
	1.5	0.08	0.09	0.00	0.00	0.19	0.20	0.05	0.04	0.17	0.18	0.13	0.13	0.08	0.07	0.17	0.18	0.00	0.00	0.09	0.09
		62	21	93	44	42	50	73	83	98	16	60	45	35	18	64	61	99	36	01	53
	2.0	0.08	0.09	0.00	0.00	0.19	0.20	0.06	0.05	0.17	0.17	0.14	0.14	0.07	0.06	0.18	0.19	0.00	0.00	0.09	0.09
		59	19	98	47	23	31	02	10	55	65	26	17	82	56	40	47	83	24	18	68
0.5L	0.4	0.08	0.09	0.00	0.00	0.19	0.20	0.05	0.04	0.18	0.19	0.12	0.12	0.09	0.08	0.15	0.15	0.01	0.01	0.07	0.08
		67	24	88	41	75	87	34	42	82	25	41	05	68	96	38	98	74	06	97	53
	0.6	0.08	0.09	0.01	0.00	0.19	0.20	0.06	0.05	0.17	0.18	0.13	0.13	0.08	0.07	0.16	0.17	0.01	0.00	0.08	0.08
		60	20	01	47	34	45	06	06	91	20	85	60	76	91	62	39	53	86	18	78
	1.0	0.08	0.09	0.01	0.00	0.18	0.19	0.07	0.06	0.16	0.16	0.16	0.16	0.07	0.06	0.18	0.19	0.01	0.00	0.08	0.09
		47	13	24	59	62	70	34	26	39	39	39	39	34	26	62	70	24	59	47	13
1.5	0.08	0.09	0.01	0.00	0.17	0.18	0.08	0.07	0.14	0.14	0.19	0.19	0.06	0.04	0.20	0.21	0.01	0.00	0.08	0.09	
	35	05	50	72	89	91	71	59	90	56	01	39	09	79	45	87	01	39	71	40	
2.0	0.08	0.08	0.01	0.00	0.17	0.18	0.09	0.08	0.13	0.13	0.21	0.21	0.05	0.03	0.21	0.23	0.00	0.00	0.08	0.09	
	24	99	72	84	31	25	87	77	74	07	15	93	20	76	78	48	87	27	87	57	

mentos de empotramiento ( $m_{AB}$  y  $m_{BA}$ ) para una trabe sometida a una carga concentrada ubicada en cualquier lugar de la viga, el modelo propuesto (MP) es el modelo matemático presentado en este

documento, donde las deformaciones por flexión y cortante se consideran, y el modelo tradicional (MT) toma en cuenta solo las deformaciones por flexión.

La Tabla 2 presenta los factores para una proporción de  $h = 0.10L$ , y la Tabla 3 muestra los factores para una relación de  $h = 0.20L$ . Estas comparaciones se realizaron con  $G = 5E/12$  para concreto,  $a = 0.2L$  y  $u/h = 1.0$ , y para  $a = 0.5L$  y  $u/h = 1.0$ , debido a que estos valores se presentan en las Tablas de la página 516 del libro de Hibbeler [19]. Los resultados que muestran las Tablas de Hibbeler es el modelo tradicional que considera las deformaciones por flexión, y el modelo propuesto toma en cuenta las deformaciones por flexión y cortante.

Otra manera para validar el modelo propuesto es como sigue: Para  $0 \leq x \leq a$  se sustituye " $u = 0h$  y  $s = 0h$ " o " $a = L$ ,  $c = 0L$  y  $u = 0h$ " en las ecuaciones (15) y (16). Para  $a \leq x \leq L - c$  se sustituye " $a = 0L$  y  $c = 0L$ " o " $u = 0h$  y  $s = 0h$ " en las ecuaciones (17) y (18). Para  $L - c \leq x \leq L$  se sustituye " $u = 0h$  y  $s = 0h$ " o " $a = 0L$ ,  $c = L$  y  $s = 0h$ " en las ecuaciones (19) y (20).

Para estas condiciones se desprecian las deformaciones por cortante. Los resultados obtenidos para los tres casos, los momentos de empotramiento son: " $M_{AB} = Pe (L-e)^2/L^2$ " y " $M_{BA} = Pe^2 (L-e) /L^2$ ". Los valores presentados anteriormente corresponden a una sección transversal constante.

Una manera de validar la continuidad de la sección transversal es la siguiente: la carga se coloca sobre los puntos " $x = a$ " y " $x = L - c$ " y usando las ecuaciones correspondientes se obtienen los mismos resultados en los momentos de empotramiento.

Por ejemplo, en las ecuaciones (15, 17) se sustituye el valor de " $e = a$ " para obtener " $M_{AB}$ " y en las ecuaciones (16, 18) para encontrar " $M_{BA}$ ", es decir, la carga concentrada se coloca sobre la unión del primero con el segundo tramo de la viga, y en las ecuaciones (17, 19) se sustituye " $e = L - c$ " para encontrar " $M_{AB}$ ", y también en las ecuaciones (18, 20) para obtener " $M_{BA}$ ", es decir, la carga concentrada se coloca sobre la unión del segundo con el tercer tramo de la viga.

Entonces el modelo propuesto en este documento es válido y no se limita para ciertas dimensiones o proporciones como algunos autores muestran, y también las deformaciones por flexión y cortante son consideradas.

## 4. Resultados

Tal como se aprecia en las Tablas 2 y 3, los factores en los momentos de empotramiento se vieron influenciados por el volumen de las cartelas en los extremos (volumen de la cartela A es  $abu/3$ , y el volumen de la cartela B es  $cbs/3$ ). A medida que se incrementa el volumen de las cartelas en el extremo "B" se observa un aumento en estos factores para el mismo apoyo y en el extremo "A" que es el apoyo opuesto se produce una disminución, esto es para los dos modelos.

Ahora de acuerdo a la comparación de ambos modelos, cuando el volumen de las cartelas es mayor en un extremo se presenta un factor mayor en los momentos de empotramiento de este apoyo para el modelo tradicional y para el extremo opuesto el mayor es el modelo propuesto.

También se observa que cuando la carga concentrada se encuentra más cerca de alguno de los apoyos la diferencia entre ambos modelos es mayor. Además, cuando las cartelas y las cargas son simétricas no se afectan los momentos de empotramiento para los dos modelos. La mayor diferencia existe para " $h = 0.2L$ ", " $a = 0.5L$ ", " $c = 0.2L$ ", " $u = h$ ", " $s = 2h$ " y " $e = 0.9L$ " en el apoyo "A" de 3.46 veces, y para " $h = 0.2L$ ", " $a = 0.2L$ ", " $c = 0.3L$ ", " $u = h$ ", " $s = 0.4h$ " y " $e = 0.1L$ " en el apoyo "B" de 2.26 veces, siendo mayor el modelo propuesto para ambos casos con respecto al modelo tradicional.

## 5. Conclusiones

En el presente trabajo se ha presentado una metodología analítica para trabes de sección transversal rectangular con cartelas parabólicas (simétricas o no simétricas) sometidas a una carga concentrada localizada en cualquier parte de la viga tomando en cuenta las deformaciones por flexión y cortante, que ha permitido determinar con precisión los factores para momentos de empotramiento. Las propiedades de la sección transversal de la viga: el ancho " $b$ " es constante y la altura " $h$ " varía a lo largo de la viga en tres partes diferentes, con variación en los tramos extremos de tipo parabólico y el tramo central recto. Las ecuaciones de compatibilidad y equilibrio se utilizaron para resolver este tipo de problema, y los

giros en los extremos de la viga se encontraron por medio del principio del trabajo virtual empleando la integraci3n exacta para obtener los factores para momentos de empotramiento.

Los modelos tradicionales consideran 6nicamente las deformaciones por flexi3n y otros autores presentan tablas que se encuentran restringidas para ciertas proporciones.

Tal como se aprecia en los resultados, los factores para momentos de empotramiento se vieron influenciados por las deformaciones de cortante. Las diferencias mayores se presentan para claros cortos.

En cualquier tipo de estructura las fuerzas cortantes y los momentos flexionantes est1n presentes; por lo tanto, aparecen las deformaciones de flexi3n y cortante. Entonces, el modelo propuesto que considera las deformaciones por flexi3n y cortante es m1s apropiado para el an1lisis estructural y tambi3n se ajusta m1s a las condiciones reales con respecto al modelo tradicional que toma en cuenta las deformaciones por flexi3n 6nicamente.

La aplicaci3n significativa de los momentos de empotramiento y las rigideces de un miembro es en los m3todos matriciales de an1lisis estructural. Los momentos de empotramiento, el factor de transporte y el factor rigidez se utilizan en el m3todo de distribuci3n de momentos (M3todo de Hardy Cross).

Adem1s, una ventaja significativa es que se pueden generar un gran n6mero de Tablas para diferentes valores de "G", "e" y proporciones de "a/L", "c/L", "h/L", "u/h" y "s/h" con ayuda de alg6n tipo de software. Las sugerencias para investigaciones futuras: 1) Cuando el miembro presenta otro tipo de secci3n transversal; 2) Cuando el miembro tiene otro tipo de configuraci3n.

## Agradecimientos

La investigaci3n descrita en este trabajo fue financiada por el Instituto de Investigaciones Multidisciplinarias de la Facultad de Contabilidad y Administraci3n de la Universidad Aut3noma de Coahuila. Los autores tambi3n agradecen a los revisores y al editor por los comentarios y sugerencias para mejorar la presentaci3n. El

estudiante de doctorado Ricardo Sandoval Rivas (CVU/Becario: 715547/590923) agradece al Consejo Nacional de Ciencia y Tecnolog1a (CONACYT) el apoyo econ3mico.

## Referencias

1. **Guldan, R. (1956).** *Estructuras aporticadas y vigas continuas.*
2. **Portland Cement Association (1958).** *Handbook of frame constants: Beam factors and moment coefficients for members of variable section.* Chicago: Portland Cement Association.
3. **Just, D.J. (1977).** Plane frameworks of tapering box and I-section. *Journal of Structural Engineering ASCE*, Vol. 103, No. 1, pp. 71–86.
4. **Schreyer, H.L. (1978).** Elementary theory for linearly tapered beams. *Journal of the Engineering Mechanics ASCE*, Vol. 104, No. 3, pp. 515–527.
5. **Medwadowski, S.J. (1984).** Nonprismatic shear beams. *Journal of Structural Engineering ASCE*, Vol. 110, No. 5, pp. 1067–182. DOI: 10.1061/(ASCE)0733-9445(1984)110:5(1067).
6. **Brown, C.J. (1984).** Approximate stiffness matrix for tapered beams. *Journal of Structural Engineering ASCE*, Vol. 110, No. 12, pp. 3050–3055. DOI: 10.1061(ASCE)0733-9445(1984)110:12(3050).
7. **Tena-Colunga, A. (2007).** *An1lisis de estructuras con m3todos matriciales.*
8. **Shooshtari, A. & Khajavi, R. (2010).** An efficient procedure to find shape functions and stiffness matrices of nonprismatic Euler-Bernoulli and Timoshenko beam elements. *European Journal of Mechanics- A/Solids*, Vol. 29, No. 5, pp. 826–836. DOI: 10.1016/j.euromechsol.2010.04.003.
9. **Yuksel, S.B. (2012).** Assessment of non-prismatic beams having symmetrical parabolic haunches with constant haunch length ratio of 0.5. *Structural Engineering and Mechanics*, Vol. 42, No. 6, pp. 849–866. DOI: 10.12989/sem.2012.42.6.849.
10. **Lu3vanos-Rojas, A. (2012).** A mathematical model for rectangular beams of variable cross section of symmetrical parabolic shape for uniformly distributed load. *Far East Journal of Mathematical Sciences*, Vol. 80, No. 2, pp. 197–230.
11. **Lu3vanos-Rojas, A. (2013).** Mechanical Elements of Rectangular Nonprismatic Members for Symmetrical Parabolic Haunches Subjected to a Uniformly Distributed Load. *Journal Architectural Engineering Technology*, Vol. 2, No. 2, pp. 1–8. DOI: 10.4172/2168-9717.1000111.

12. **Luévanos-Rojas, A. & Montoya-Ramírez, J. (2014).** Mathematical model for rectangular beams of variable cross section of symmetrical linear shape for uniformly distributed load. *International Journal of Innovative Computing, Information and Control*, Vol. 10, No. 2, pp. 545–564.
13. **Luévanos-Rojas, A., Luévanos-Rojas, R., Luévanos-Soto, I., Luévanos-Vázquez, R.G., & Ramírez-Luévanos, O.A. (2014).** Mathematical Model for Rectangular Beams of Variable Cross Section of Symmetrical Linear Shape for Concentrated Load. *International Journal of Innovative Computing, Information and Control*, Vol. 10, No. 3, pp. 851–881.
14. **Luévanos-Rojas, A. (2014).** A mathematical model for fixed-end moments for two types of loads for a parabolic shaped variable rectangular cross section. *Ingeniería e Investigación*, Vol. 34, No. 2, 17–22. DOI: 10.15446/ing.investig.v34n2.44705.
15. **Luévanos-Rojas, A. (2015).** Modelado para vigas de sección transversal “I” sometidas a una carga uniformemente distribuida con cartelas rectas. *Ingeniería Mecánica Tecnología y Desarrollo*, Vol. 5, No. 2, pp. 281–292.
16. **Luévanos-Soto, I. & Luévanos-Rojas, A. (2017).** Modeling for fixed-end moments of I-sections with straight haunches under concentrated load. *Steel and Composite Structures*, Vol. 23, No. 5, pp. 597–610. DOI: 10.12989/scs.2017.23.5.597.
17. **Luévanos-Rojas, A., López-Chavarría, S., & Medina-Elizondo, M. (2016).** Modeling for mechanical elements of rectangular members with straight haunches using software: part 1. *International Journal of Innovative Computing, Information and Control*, Vol. 12, No. 3, pp. 973–985.
18. **Luévanos-Rojas, A., López-Chavarría, S., & Medina-Elizondo, M. (2016).** Modeling for mechanical elements of rectangular members with straight haunches using software: part 2. *International Journal of Innovative Computing, Information and Control*, Vol. 12, No. 4, pp. 1027–1041.
19. **Hibbeler, R.C. (2006).** *Structural analysis*. New Jersey: Prentice-Hall, Inc.
20. **Vaidyanathan, R. & Perumal, P. (2006).** *Structural Analysis*. New Delhi: Laxmi Publications (P) LTD.
21. **Williams, A. (2009).** *Structural Analysis: In Theory and Practice*. New York: Butterworth Heinemann.
22. **McCormac, J.C. (2007).** *Structural Analysis: using classical and matrix methods*. New York: John Wiley & Sons.
23. **Ghali, A., Neville, A.M., & Brown, T.G. (2003).** *Structural Analysis: A Unified Classical and Matrix Approach*. New York: Taylor & Francis.
24. **González-Cuevas, O.M. (2007).** *Análisis Estructural*. México: Limusa.
25. **Gere, J.M. & Goodo, B.J. (2009).** *Mechanics of Materials*. New York: Cengage Learning.

Article received on 02/01/2018; accepted on 08/01/2019.  
Corresponding author is Arnulfo Luévanos Rojas.

# A Numerical Perspective on the Jaynes-Cummings Model Wigner Function

J. C. García-Melgarejo<sup>1</sup>, N. Lozano-Crisóstomo<sup>1</sup>, J. Jesús Escobedo-Alatorre<sup>2</sup>, K. J. Sánchez-Pérez<sup>3</sup>,  
M. Torres-Cisneros<sup>4</sup>, E. S. Arroyo-Rivera<sup>3</sup>, R. Guzmán-Cabrera<sup>4</sup>

<sup>1</sup> Universidad Autónoma de Coahuila,  
Facultad de Ingeniería Mecánica y Eléctrica, Torreón,  
Mexico

<sup>2</sup> Universidad Autónoma del Estado de Morelos,  
Centro de Investigación en Ingeniería y Ciencias Aplicadas, Cuernavaca,  
Mexico

<sup>3</sup> Instituto Nacional de Astrofísica, Óptica y Electrónica, Tonantzintla,  
Mexico

<sup>4</sup> Universidad de Guanajuato,  
Departamento de Ingeniería Electrónica, Guanajuato,  
Mexico

{julio.melgarejo, n.lozano}@uadec.edu.mx, jescobedo@uaem.mx,  
{kjaneth279, earroyorivera}@inaoep.mx, {torres.cisneros, guzmanc}@ugto.mx

**Abstract.** The Wigner function appeared to discuss the quantum correction of thermodynamic equilibrium, and it has become a tool for the analysis of quantum systems, especially the harmonic oscillator, which states describe the quantum field in a cavity. We discuss a matrix approach for the computation of the Wigner function. The numerical techniques here discussed are applied to obtain the time-dependent Wigner function of the field for the Jaynes-Cummings Model, which is widely known to describe the fundamental matter-field interactions in a perfect cavity.

**Keywords.** Wigner function, Jaynes-Cummings model.

## 1 Introduction

According to the foundations of the quantum mechanics, the complete description of a quantum system relies on the knowledge of the state vector  $\psi$  that belongs to the space states [5]. However, it is difficult to extract, in a transparent manner, the most relevant physical insights of a quantum

system through the single analysis of its vector state. Fortunately, several useful representations can demystify its abstract nature, among them we find the Wigner function. It is primordially appropriated to discuss the connection between the classical and the quantum domain [15].

The initial emergence of the field Wigner function in cavities has become increasingly important as fundamental Quantum question has received a renewed attention and that demanded of a handy theoretical, numerical and experimental tool able, as the Wigner function, to express its analysis. This is particularly important in composite quantum systems [11], where its complexity limits quite frequently our ability to provide fully solvable results.

In those systems, the Wigner function becomes far too complex to be solved analytically and the need to have a fully numerical approach is required. The aim of this work is to introduce

a numerical approach that can explore from its basics.

Several equivalent definitions allow the computation of the Wigner function. If it is known the quantum system stationary wave function  $\phi(x)$ , we can easily compute the Wigner function through the following expression [15]:

$$W(x, p) = \frac{1}{2\pi\hbar} \int d\xi \exp\left(-\frac{i}{\hbar} p\xi\right) \phi^*\left(x - \frac{1}{2}\xi\right) \phi\left(x + \frac{1}{2}\xi\right). \quad (1)$$

An alternative and equivalent way to compute the Wigner function is through [3]:

$$W(\alpha, \alpha^*) = 2\text{Tr} \left[ \rho D(\alpha) e^{i\pi a^\dagger a} D^{-1}(\alpha) \right]. \quad (2)$$

Let us notice that while the equation 1 is valid for a system described by the wave function  $\phi(x)$ , the equation 2 is of more general application because applies to a quantum system described by the density operator  $\rho$ .

Maybe, the most widely discussed model in quantum optics is the Jaynes-Cummings Model (JCM), which in its simplest form describes the interactions between a two-level atom and the quantized field in a perfect cavity [10]. The analysis of the field Wigner function in this model is a standard tool to inquire on the properties of the cavity quantum field. To compute the JCM Wigner function, it has been proposed the factorization of JCM the wave function as:

$$|\phi(t)\rangle = |\phi_{-1}(t)\rangle |-1\rangle + |\phi_1(t)\rangle |1\rangle. \quad (3)$$

is a key step in the discussions of the analytical time-dependent JCM Wigner function. The limitation of the equation 3 is its validity only for fields described by a vector state, and it is desirable to generalize such expression to take into account a density matrix formalism. Such a handy factorization has been possible through the assumption of exact atom-field resonance [6].

In the equation 3, the first term is related to the ground state and the second one to the excited state. An analysis without those assumptions becomes quite a formidable task, which implicitly makes desirable numerical strategies that could provide an efficient analysis to more complex cases. This numerical perspective is the motivation

of our didactic examination of the JCM Wigner function as a convenient example, which can be extended to discuss many other problems.

The starting point of our discussion will be the equation 2. The computational implementation of the quantum operators, which appear in such equation, becomes viable by taking advantage of the modern software available for the matrix management. The quantum practitioner, which often finds these complex operator expressions, will appreciate it to explore these techniques in many other problems further.

## 2 The Harmonic Oscillator Wigner Function

The Harmonic Oscillator (HO) is one of the most widely used models in many areas of physics. The quantum version of this model is the mathematical tool to describe the quantum field inside a perfect cavity. Therefore, before inquiring into the JCM cavity field Wigner function, it is convenient to briefly describe the quantum harmonic oscillator and to show the the expressions of its time-dependent Wigner function. The Hamiltonian of the quantum harmonic oscillator is:

$$H_{\text{HO}} = \hbar\omega \left( a^\dagger a + \frac{1}{2} \right), \quad (4)$$

where its frequency is given by  $\omega$  and the rising and lowering operators are provided by  $a^\dagger$  and  $a$  respectively, and their properties have a vast literature.

There are quantum states expressed in terms of  $a$  and  $a^\dagger$ , as the mathematical tool to describe the quantum field in cavities. The Fock state  $|n\rangle$  is defined through the eigenvalue relation  $a^\dagger a |n\rangle = n |n\rangle$ , where  $n$  denotes an integer number. The Fock states are known for having a well-defined number of quanta and for not having a classical counterpart. These states are particularly relevant since they allow a discrete representation of quantum states and operators, which is a convenient way for its numerical implementation. The coherent states, introduced by Glauber [7], are defined as the eigenstate of the annihilation



operator  $a|\alpha\rangle = \alpha|\alpha\rangle$ , where  $\alpha$  is, in general, a complex number. They have the closest classical-like behavior, and they describe the laser beam light accurately.

The HO propagator is just the exponential of the Hamiltonian  $\exp(-iHt/\hbar)$ :

$$U_{\text{HO}}(t) = \exp(-i\omega a^\dagger a t). \quad (5)$$

Moreover, the HO state at an arbitrary time  $t$  is:

$$|\phi(t)\rangle = U_{\text{HO}}(t)|\phi(0)\rangle. \quad (6)$$

Two typical examples of propagating an initial state are the Fock state  $|n\rangle$ , which at  $t$  acquires only a phase factor  $e^{-in\omega t}|n\rangle$ . On the other hand, if the HO initially is a coherent state, the state becomes  $|\alpha e^{-i\omega t}\rangle$ . The corresponding analytical expression of the Wigner function is:

$$W(\alpha, \alpha^*) = 2(-1)^n e^{-4|\alpha|^2} L_n(2|\alpha|^2), \quad (7)$$

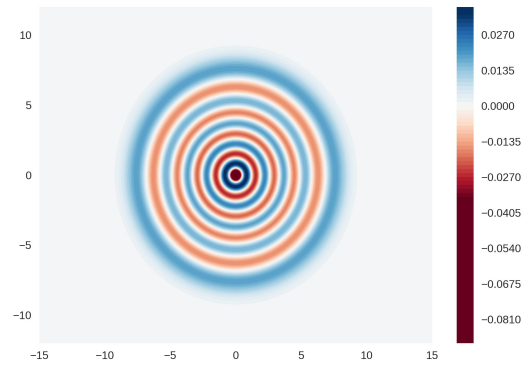
for a Fock state. In equation (1.7)  $n$  denotes the HO number of quanta, and  $L_n(x)$  denotes the  $n$ th-order Laguerre polynomial. Notice that this Wigner function is time-independent because in this case the density matrix does not depend explicitly on time, see Fig. 1. The Wigner function of a coherent state is given by:

$$W(\alpha, \alpha^*) = 2e^{-|\alpha - \alpha_0(t)|^2}. \quad (8)$$

The equation 8 describes a clockwise rotating two-dimensional Gaussian function in the complex plane defined by  $\alpha$ , which is centered at  $\alpha_0(t)$ . Both didactic examples 7 and 8, as well the Wigner function of other remarkable states, are well-known and are reported in the literature [14]. Despite the simplicity of the analytical expressions 7 and 8, the computations are lengthy, even under this simple Hamiltonian.

### 3 The Jaynes-Cummings Model

The Jaynes-Cummings Model (JCM) [10] is one of the most notorious physical models to accurately describe the interactions between the quantized field in a one-dimensional perfect cavity of frequency  $\omega$  and a Two-Level Atom (TLA) with



**Figure 1.** Numerical evaluation of the Fock state Wigner function of a harmonic oscillator prepared with 9 photons. This Wigner function has negative zones, indicating it is a non-classical state and it is time-independent because the evolution of the Fock state is determined only by a phase factor

atomic transition frequency  $\omega_0$ . The strength of such interactions are provided by the coupling constant  $\lambda$ . The JCM Hamiltonian is:

$$H = \frac{1}{2}\hbar\omega_0\sigma_z + \hbar\omega a^\dagger a + \hbar\lambda(a\sigma_+ + a^\dagger\sigma_-). \quad (9)$$

The above Hamiltonian was obtained under Rotating Wave Approximation (RWA). Just like we anticipated, the operator  $a$  describes the cavity mode, and satisfies the usual Bosonic operators commutation rule  $[a^\dagger, a] = 1$ . On the other hand, the atomic rising  $\sigma_+ = |1\rangle\langle-1|$  and lowering  $\sigma_- = |-1\rangle\langle 1|$  operators govern the transitions between the atomic excited  $|1\rangle$  and ground state  $|-1\rangle$ . They are related to the atomic inversion operator through the commutation rule  $\sigma_z = [\sigma_+, \sigma_-]$ .

There are several outstanding mathematical properties of this model. One of them is the existence of motion constants. They are the total number of excitation and the interchange constant [1], which are given by:

$$N = a^\dagger a + \frac{1}{2}(\sigma_z + 1), \quad (10a)$$

$$C = \frac{1}{2}\Delta\sigma_z + \lambda(a\sigma_+ + a^\dagger\sigma_-). \quad (10b)$$

Often the computation of JCM analytical expressions relies on the knowledge of these constants.

Furthermore, the technique of finding the motion constants is quite a useful path for solving more complex quantum-electrodynamical systems. The propagator of the JCM is given by:

$$U(t) = \exp(-i\omega Nt) \exp(-iCt). \quad (11)$$

Another JCM feature is the existence of the so-called dressed states that diagonalize the Hamiltonian, and are given by:

$$|\psi_0\rangle = |-1, 0\rangle, \quad (12a)$$

$$|\psi_n^+\rangle = c_n |1, n-1\rangle + s_n |-1, n\rangle, \quad (12b)$$

$$|\psi_n^-\rangle = -s_n |1, n-1\rangle + c_n |-1, n\rangle. \quad (12c)$$

In the equation 12 we are going to choose the Bosonic index  $n$  satisfying the condition  $n > 0$ . The quantities  $c_n$  and  $s_n$  are a shorthand notation for  $c_n = \cos(\theta_n/2)$  and  $s_n = \sin(\theta_n/2)$ , where  $\theta_n$  is defined through:

$$\cos \theta_n = \frac{\Delta}{\sqrt{\Delta^2 + 4\lambda^2 N_{nm}}}, \quad (13a)$$

$$\sin \theta_n = \frac{2\lambda\sqrt{N_{nm}}}{\sqrt{\Delta^2 + 4\lambda^2 N_{nm}}}. \quad (13b)$$

The dressed states are a complete base for the JCM. Therefore, a JCM unity operator is the following:

$$I_{\text{JCM}} = |\psi_0\rangle\langle\psi_0| + \sum_n \sum_{\xi=+,-} |\psi_n^\xi\rangle\langle\psi_n^\xi|. \quad (14)$$

The exact JCM eigenfrequencies are [13]:

$$\omega_{nm}^\pm = N_{nm}\omega + \frac{1}{2} \left( \Delta \pm \sqrt{\Delta^2 + 4\lambda^2 N_{nm}} \right). \quad (15)$$

The total excitation was denoted by  $N_{nm}$ , and it is given by:

$$N_{nm} = n + \frac{1}{2}(m+1), \quad (16)$$

where  $n$  and  $m$  are the eigenvalues of the unperturbed operators  $a^\dagger a$  and  $\sigma_z$  respectively.

## 4 Numerical Implementation of Quantum States and Operators

As we have previously pointed out, the time-dependent cavity field Wigner function computation is a formidable task, even in the absence of atomic interactions. However, the definition given in equation 2, provides quite a convenient manner to numerically perform such a task. For this purpose, we have to recall that the quantum states and operators have computer readable representations. In this matrix formulation of the quantum mechanics, a quantum state is described by a column vector while a quantum operator by a matrix. In general, both of them have complex entries.

### 4.1 Atomic Operators and States

Most of the modern software for matrix management allow the implementation of column vectors and several other helpful matrix operations, like the matrices Kronecker product or the matrix exponentiation. The excited  $|1\rangle$  and the ground  $|-1\rangle$  atomic states are represented through the following column vectors:

$$|1\rangle = \begin{pmatrix} 1 \\ 0 \end{pmatrix}, \quad (17)$$

and

$$|-1\rangle = \begin{pmatrix} 0 \\ 1 \end{pmatrix}. \quad (18)$$

On the other hand, the rising and the lowering atomic operators have the matrix representation:

$$\sigma_+ = \begin{pmatrix} 0 & 1 \\ 0 & 0 \end{pmatrix}, \quad (19)$$

and

$$\sigma_- = \begin{pmatrix} 0 & 0 \\ 1 & 0 \end{pmatrix}. \quad (20)$$

The dimension of the vectors is  $1 \times 2$ , while the dimension of the operators is  $2 \times 2$  due to the two-dimensional TLA Hilbert space.

## 4.2 Field Operators and States

Field states and operators also have to be implemented on the computer. The Fock  $|n\rangle$  state, and the operators of annihilation and creation, have an infinite dimension. Therefore, a knowledgeable practitioner has to approximate them. Let us choose a convenient example of truncation at  $n_{\max} = 5$ . We have to establish a criterion to obtain an upper bound to describe numerically these operators. Often this upper bound is dictated in terms of state distribution mean number, see Fig. 1. A field described by a Fock state with two photons,  $n = 2$ , is approximated by the column vector:

$$|2\rangle = \begin{pmatrix} 0 \\ 0 \\ 1 \\ 0 \\ 0 \end{pmatrix}. \quad (21)$$

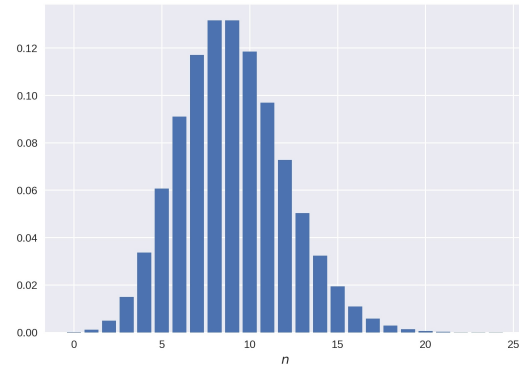
In the same terms, the coherent state can be easily approximated with the following equation:

$$|\alpha\rangle = e^{-|\alpha|^2/2} \sum_{n=0}^{n_{\max}} \frac{\alpha^n}{n!} |n\rangle, \quad (22)$$

where  $\alpha$  is a complex number. The annihilation operator, in the same terms, has the matrix representation:

$$a = \begin{pmatrix} 0 & \sqrt{1} & 0 & 0 & 0 \\ 0 & 0 & \sqrt{2} & 0 & 0 \\ 0 & 0 & 0 & \sqrt{3} & 0 \\ 0 & 0 & 0 & 0 & \sqrt{4} \\ 0 & 0 & 0 & 0 & 0 \end{pmatrix}. \quad (23)$$

In the same analogous terms,  $a^\dagger = (a)^T$  due to we have chosen the real representation for the  $a$  matrix.



**Figure 2.** Numerical evaluation of the photon counting probability, i.e., the first 25 diagonal elements of the density matrix  $|\alpha_0\rangle\langle\alpha_0|$ . The average number of this distribution has been chosen to be  $|\alpha_0|^2 = 9$ . The sum of all these diagonal elements is the total probability and it is numerically equal to 0.999991346873. This tell us that with this number of elements, we have an excellent description of this particular quantum state..

## 4.3 Interacting Systems

The last two sections describe how to implement the states and operators of two isolated systems, the atom and the field. However, just like the JCM establish, these two systems are interacting. In that case, the extended Hilbert space  $\mathcal{H} = \mathcal{H}_{\text{TLA}} \otimes \mathcal{H}_{\text{FIELD}}$  provides the mathematical description of the JCM dynamics, where  $\mathcal{H}_{\text{TLA}}$  and  $\mathcal{H}_{\text{FIELD}}$  denote the Hilbert space of each subsystem. To work in the extended Hilbert space  $\mathcal{H}$ , we must implement extensions of the isolated operators as well as the quantum states [4]. Numerically, both cases are implemented through the Kronecker product of two matrices, which for two arbitrary matrices  $A = \{a_{ij}\}$  and  $B = \{b_{ij}\}$  is given by [9]:

$$A \otimes B = \begin{pmatrix} a_{11}B & \dots & a_{1n}B \\ \vdots & \ddots & \vdots \\ a_{m1}B & \dots & a_{mn}B \end{pmatrix}. \quad (24)$$

If we restrict ourselves to the JCM, the initial state will be given by  $|\psi_0\rangle = |\psi_{\text{TLA}}\rangle \otimes |\psi_{\text{FIELD}}\rangle$  the dimension of the resulting discrete Hilbert space is

$2 \times n_{\max}$ . The extension of the atomic operators  $O_{\text{TLA}}$  are given by:

$$\tilde{O}_{\text{TLA}} = O_{\text{TLA}} \otimes I_{\text{FIELD}}. \quad (25)$$

The extension of an arbitrary field operator  $O_{\text{FIELD}}$  is given by:

$$\tilde{O}_{\text{FIELD}} = I_{\text{TLA}} \otimes O_{\text{FIELD}}. \quad (26)$$

In both equations (1.25) and (1.26),  $I_{\text{FIELD}}$  and  $I_{\text{TLA}}$  denote the field and the TLA identity operators respectively. The operators  $\tilde{O}_{\text{TLA}}$  and  $\tilde{O}_{\text{FIELD}}$  are matrices that have the same dimensions and obey the standard matrix operations.

#### 4.4 Numerical Solution

The proper implementation of the extended operators in the JCM framework allows an easy matrix construction of Hamiltonian using typical computer matrix operations. Such computer implementation is advantageous because it allows knowing the state of the JCM at an arbitrary time through several methods. For instance, through the numerical solution of the first-order system of differential equations, provided by the Schrödinger equation:

$$i\hbar \frac{d|\phi(t)\rangle}{dt} = H|\phi(t)\rangle. \quad (27)$$

The numerical techniques to solve differential equations systems, as the given in equation 27, are widely known as are its limits, precision, and convergence [2, 8]. Solving numerically the system 27 is a very general approach, which however has its practical limitations that can be overcome in particular cases like the JCM. A second alternative approach is given by expressing equation 11 in our matrix representation:

$$U(t) = \exp(-i\omega Nt) \exp(-iCt), \quad (28)$$

Where  $N$  and  $C$  are the discrete versions of the operators defined in 10. A third alternative, a more convenient for our purposes, is expanding the wave function in terms of the numerical dressed states  $H|\phi_k\rangle = E_k|\phi_k\rangle$  and its eigenstates, which are numerically available:

$$|\phi(t)\rangle = U(t) \sum_{k=0}^{2 \times n_{\max}} |\psi_k\rangle \langle\psi_k|\phi(0)\rangle \quad (29a)$$

$$= \sum_{k=0}^{2 \times n_{\max}} c_k \exp(-i\omega_k t) |\psi_k\rangle. \quad (29b)$$

The coefficient  $c_k$  is easily computed through the matrix operation  $c_k = \langle\psi_k|\phi(0)\rangle$ , as well as the numerical eigenfrequencies that are given by  $\omega_k = E_k/\hbar$ . The vector state at  $t$ , computed by either of the mentioned methods, leads to the direct computation of the JCM density matrix  $\rho(t) = |\phi(t)\rangle \langle\phi(t)|$ , i.e., to the knowledge of the physical properties of the JCM, in particular the Wigner function.

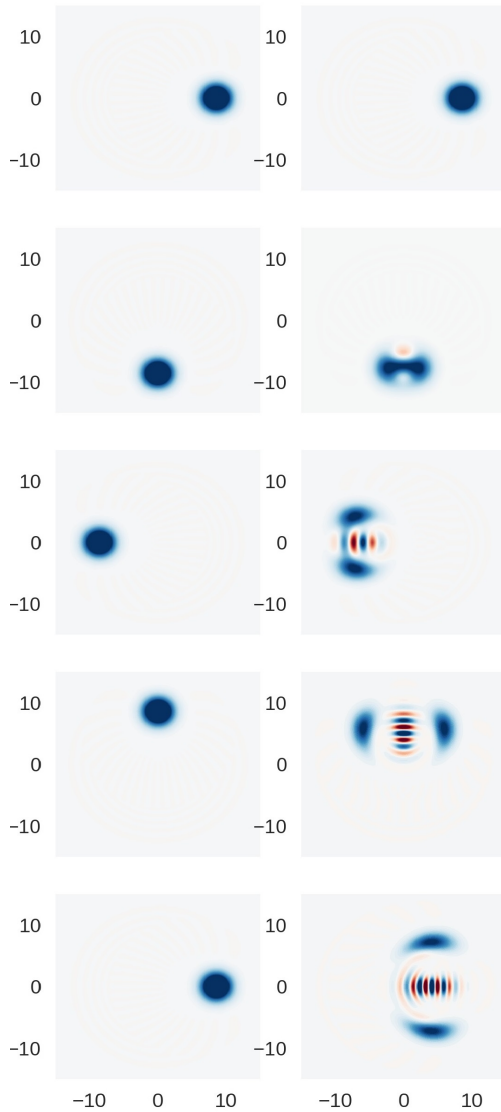
#### 5 Numerical Computation of the JCM Wigner Function

The matrix formulation of the quantum operators that we provided is intended to describe interacting atom-field systems. However, the definition of Wigner function requires only the density matrix of the field. In addition to the Kronecker product, we have to implement an operation that allows reducing the dimension of the matrix of a composite system to obtain only the density matrix of the field. This operation is called partial trace.

Let us discuss the concept of the trace. Consider an arbitrary base  $|\chi_i\rangle$  that belongs to a Hilbert space  $\mathcal{H}$ , and also an operator  $O$  living in a space denoted by  $\mathcal{L}(\mathcal{H})$ . The trace of  $O$  is the sum of the diagonal elements in the mentioned base:

$$\text{Tr}\{O\} = \sum_{i=1}^d \langle\chi_i|O|\chi_i\rangle, \quad (30)$$

where  $d$  is the dimension of  $\mathcal{H}$ . The trace is, in general, a complex quantity and may be taken in another basis which includes those with continuous indices. The JCM density matrix  $\rho_{\text{JCM}}(t)$  is described in the extended Hilbert space  $\mathcal{H} = \mathcal{H}_{\text{TLA}} \otimes \mathcal{H}_{\text{FIELD}}$ . In consequence, we must reduce the dimension of  $\rho_{\text{JCM}}$  by dropping the atomic elements, this procedure is known as partial trace



**Figure 3.** Comparison between the numerical time-dependent Wigner function of a harmonic oscillator (On the left) and the resonant JCM (On the right). In both cases, the initial field is coherent with an average number of photons equal to nine. Consistently with the equation (1.8), the coherent state of the oscillator rotates clockwise keeping its shape. The coherent state in the JCM splits into two contributions, and in the middle, there is exhibit quantum interference that can be interpreted as an atom-field entanglement. Also it is observed a clockwise rotation, which it is reminiscent of the non-interacting part of the JCM Hamiltonian. For this numerical experiment, the TLA was prepared in the excited state

and its denoted by  $\rho_{\text{FIELD}} = \text{Tr}_{\text{TLA}} \{ \rho_{\text{JCM}}(t) \}$ . It is implemented through [12]:

$$\rho_{\text{FIELD}} = \sum_{i=-1,1} (I_{\text{FIELD}} \otimes \langle i|) \rho_{\text{JCM}} (I_{\text{FIELD}} \otimes |i\rangle). \quad (31)$$

On the other hand, if we are interested in the atomic density matrix -for instance, to inquire in the Bloch vector dynamics-, we can follow an analog definition to obtain the TLA density matrix:

$$\rho_{\text{TLA}} = \sum_{j=0}^{n_{\text{max}}} (\langle j| \otimes I_{\text{TLA}}) \rho_{\text{JCM}} (|j\rangle \otimes I_{\text{TLA}}). \quad (32)$$

With the knowledge of the density matrix, we can easily implement a routine to compute the Wigner function:

$$W(\alpha, \alpha^*) = 2 \text{Tr} \left[ \rho_{\text{FIELD}} D(\alpha) e^{i\pi a^\dagger a} D^{-1}(\alpha) \right], \quad (33)$$

where  $D(\alpha)$  is the displacement operator, which is given by:

$$D(\alpha) = \exp(\alpha^* a - \alpha a^\dagger). \quad (34)$$

The numerical implementation of  $D(\alpha)$  and  $e^{i\pi a^\dagger a}$  can be straightforwardly done in most of the modern software for matrix management. To become computationally more efficient, we can use the cyclic property of the trace and take the trace, by using Fock states, to obtain an expression that requires fewer matrix exponentiations:

$$W(\alpha, \alpha^*) = 2 \sum_{n=0}^{n_{\text{max}}} (-1)^n \langle n| D(-\alpha) \rho_{\text{FIELD}} D(\alpha) |n\rangle. \quad (35)$$

The matrix operations involved in the equation 35 makes clear the power of the matrix methods for the computation of the Wigner function.

## 6 Final Remarks

The numerical techniques here developed can be used to extend easily the results presented in Fig.3. Among these extensions, we can consider the cavity field prepared with other coherence properties. This is done by just changing the initial

state of the field  $\rho_{\text{FIELD}}$ . Also, we can explore more complex cases, for instance, the time-dependent Wigner function for the non-resonant JCM, or a TLA prepared in a superposition of excited and ground state. Despite we applied the formalism to the numerical computation of the Wigner function, we also can use it to inquire into other quantities of interest such as expectation values of  $\sigma_z$ , widely known as the atomic inversion.

In the matrix approach here presented, the most relevant source of numerical errors is introduced by the approximated field operators and its posterior exponentiation. To guarantee a good approximation, we have to take the dimension of the approximated field operators  $n_{\text{max}}$  large enough; the criterion followed in this work is to select  $n_{\text{max}}$  that makes the sum of the diagonal elements very close to 1. Let us recall that in Fig.1 we choose the value 0.999991346873.

There are several matrix management software. Matlab is an excellent option, but also there are free options. For instance numy and scipy, which has the python programming language at its core. These libraries also have capabilities to manage sparse matrices, which can reduce considerably the time of execution of the numerical routines required for the matrix operations in quantum problems.

### Acknowledgments.

J. C. García-Melgarejo and N. Lozano-Crisóstomo thank to the Universidad Autónoma de Coahuila (UAdeC) and to the Instituto Nacional de Astrofísica, Óptica y Electrónica (INAOE) all the support that made possible the realization of this work.

### References

1. Ackerhalt, J. & Rzazewski, K. (1975). Heisenberg-picture operator perturbation theory. *Phys. Rev. A* 12 (6).
2. Butcher, J. (2016). *Numerical methods for ordinary differential equations*, chapter Numerical Differential Equation Methods. Wiley, pp. 51–133.
3. Cahill, K. & Glauber, R. (1969). Ordered expansions in boson amplitude operators. *Physical Review* 177 (5).
4. Cohen-Tannoudji, C., Bernard, D., & Frank, L. (1991). *Quantum Mechanics, Vol. 1*, chapter The mathematical tools of quantum mechanics. Wiley, pp. 91–209.
5. Dirac, P. A. M. (1981). *The principles of quantum mechanics*, chapter Dynamical variables and observables. Oxford University Press, pp. 23–49.
6. Gerry, C. & Knight, P. (2005). *Introductory quantum optics*, chapter Emission and absorption of radiation by atoms. Cambridge University Press, pp. 74–114.
7. Glauber, R. (1963). Coherent and incoherent states of the radiation field. *Physical Review* 131 (6).
8. Hochbruck, M. & Ostermann, A. (2010). Exponential integrators. *Acta Numerica* 19.
9. Horn, R. & Johnson, C. (1991). *Topics in Matrix Analysis*, chapter Matrix equations and the Kronecker product. Cambridge University Press, pp. 239–288.
10. Jaynes, E. & Cummings, F. (1963). Comparison of quantum and semiclassical radiation theories with application to the beam maser. *Proc. IEEE*. 51 (1) : 89–109, IEEE.
11. Marchiolli, M., Missori, R., & Roversi, J. (2003). Qualitative aspects of entanglement in the jaynes-cummings model with an external quantum field. *Journal of Physics A: Mathematical and General* 36 (49).
12. Maziero, J. (2017). Computing partial traces and reduced density matrices. *Int. J. Mod. Phys. C* 28.
13. Narozhny, N., Sanchez-Mondragon, J., & Eberly, J. (1981). Coherence versus incoherence: Collapse and revival in a simple quantum model. *Phys. Rev. A* 23 (1).
14. Schleich, W. P. (2001). *Quantum optics in phase space*, chapter Quantum states in phase space. Wiley, pp. 99–144.
15. Wigner, E. (1932). On the quantum correction for thermodynamic equilibrium. *Phys. Rev.* 40 (5).

Article received on 24/10/2018; accepted on 15/02/2019.  
Corresponding author is J. C. García-Melgarejo.