ISSN 2007-9737

Keyphrase Extraction: The Open Problem of Algorithm Comparability and a Path toward its Resolution

Svetlana Popova*

Technical University of Dublin, Dublin,
Ireland

spbu.svp@gmail.com

Abstract. Keyphrases provide a compact representation of a document's content. Being more descriptive than single keywords, keyphrases help to perform efficient text mining. Keyphrase extraction remains a difficult problem due to the limited performance of existing algorithms. Developing new approaches requires a robust framework for evaluating proposed algorithms. However, almost no attention is given to the validity of comparisons in the domain. There is a lack of studies that comprehensively summarize the evaluation issues in the keyphrase extraction domain. This research aims to bridge this gap and systematize the existing differences in F1-score calculation, which is the most popular evaluation score in the domain. We demonstrate the extent to which the macro-average F1-score can vary when calculated for the same algorithms and datasets but in different manners. We propose the 'Q-10' evaluation pipeline, which consists of ten questions about the evaluation process that should be answered to correctly compare results across different algorithms. We present the quality scores for unsupervised keyphrase extraction algorithms obtained using several F1-score calculation methods. Based on the 'Q-10' evaluation pipeline and cross-paper comparisons, we compare supervised RNN- and transformer-based approaches with unsupervised algorithms. The study involves 10 unsupervised algorithms and 6 well-known datasets.

Keywords. Keyphrase extraction, evaluation problem, F1 scoring problem, correct comparison of algorithms.

1 Introduction and Related Work

Keyphrases offer a concise summary of the main content of a document [28], and the goal of automatic keyphrase extraction **(KE)** is the automatic selection of important and topical phrases from the body of a document [32]. Effective keyphrases should meet certain criteria [17]: they should be meaningful, relevant, and provide broad coverage. Compared to individual keywords, keyphrases give a richer description, which enhances the efficiency of text mining and information retrieval [10, 2, 37]. The performance of existing KE algorithms remains far from being high, and the development of new approaches requires objective evaluation and comparison of algorithms. This paper examines the key issues involved in such an evaluation.

We will distinguish keyphrase extraction and keyphrase generation. In the keyphrase extraction task, it is assumed that extracted phrases are presented in the text. In keyphrase generation, the goal is to obtain phrases that are assigned to the text by a human but do not appear as identical word sequences in the text. We only deal with the keyphrase extraction problem and focus primarily on unsupervised methods; however, the evaluation issues highlighted in the research are also relevant to supervised methods.

This study is limited to datasets from the scientific domain, but the evaluation issues discussed for algorithm comparison extend to other domains.

In datasets for KE evaluation, humans assigned keyphrases to each text. The KE algorithm's task is to extract from the text the same phrases that were manually assigned. One of the primary methods for evaluation in the KE domain is exact-match F1-score. Although it is not the only method

used, and although other evaluation approaches have been proposed over time, the F1-score has become foundational and is used in almost all publications. Most algorithms are compared using the F1-score, but the authors do not assess the validity of these comparisons.

This question has only recently begun to be addressed [9] where authors noted: "... three papers about keyphrase extraction published in the same year at ACL, since neither benchmark datasets, parameter settings nor evaluation metrics are comparable". There is a lack of studies that comprehensively summarize the evaluation issues to be considered when comparing KE algorithms.

We aim to partially bridge the F1 evaluation gap and systematize the existing differences in how this score is calculated. We report evaluation results for 10 algorithms on 6 datasets, using several different ways to calculate the macro-average F1-score.

This allows us to show how F1-score values for the same algorithms and datasets vary depending on how the score is calculated. This shows that we need to examine which factors affect the F1 score and standardize them when calculating F1 for comparable algorithms. In this paper, we propose a way to achieve this standardization—the "Q-10" evaluation pipeline.

2 Problem Setting

2.1 Keyphrase Extraction

Keyphrase Extraction is defined as follows. Let $D = \{d_1, d_2, ..., d_n\}$ be a set of n documents. Each document $d_i \in D$ has reference phrases (a gold standard) – a set of keyphrases predefined by the experts $C_i = \{c_{i_1}, c_{i_2}, ..., c_{i_m}\}$. The goal of an unsupervised keyphrase extraction approach is for each text $d_i \in D$ automatically extract a set of k keyphrases $G_i = \{g_{i_1}, g_{i_2}, ..., g_{i_k}\}$ that should match the set of reference phrases (gold-standard phrases) as precisely as possible in terms of evaluation scores, e.g. exact-match Precision, Recall and F1-score (F1) (described below).

2.2 Evaluation with F1-score

F1-score is the main evaluation score in the KE domain. The notation F1-score@k (F1@k) is used to evaluate an algorithm when its task is to extract at most k keyphrases per text.

F1-score can be calculated for a dataset as a micro- or macro-average:

$$Precision = \frac{|(C \cap G)|}{|G|} \tag{1}$$

$$Recall = \frac{|(C \cap G)|}{|C|} \tag{2}$$

$$F1 - score = \frac{2 \times Precision \times Recall}{Precision + Recall} \tag{3}$$

Where:

— For micro-average F1-score: $|C \cap G|$ - is the number of correctly extracted phrases when processing all the texts of the collection, $|G|$ - is the total number of phrases automatically extracted by the algorithm from all the texts of the collection, $|C|$ - is the number of all phrases in the gold standard.

— For macro-average F1-score Precision, Recall and F1-score are calculate for each text and the final value of F1-score for dataset is taken as mean value of F1-score for texts from this dataset. Here for every text $|C \cap G|$ - is the number of correctly extracted phrases, $|G|$ - is the number of phrases automatically extracted by the algorithm, $|C|$ - is the number of phrases in the gold standard for the text.

The evaluation results obtained using micro- and macro-average F1-scores differ in their values.

3 Factors Contributing to Discrepancies in Quality Evaluation

3.1 Comparative Approaches for Algorithm Evaluation and Associated Challenges

To compare algorithms in the KE domain, two approaches are commonly used:

— cross-paper comparison of results;

— re-implementation of the compared algorithms to evaluate their performance.

Authors compare the F1-scores obtained for their algorithms with the F1-scores obtained for other algorithms. However, in many papers, authors simply state that they use the F1-score for evaluation without explaining exactly how it was calculated. Both cases mentioned above have nuances that must be considered for the correct comparison of evaluation scores:

— consistency of settings across algorithms;

— uniformity in the method used to calculate the F1-score.

Both positions should be carefully considered to ensure an accurate comparison of the evaluation scores of different algorithms. Below, we will list the factors leading to differences in the F1-score.

3.2 Main Issues in Keyphrase Evaluation

Outline the main issues that influence F1-score evaluation results.

3.2.1 Differences in the Processing of Dataset References

The problems listed below are relevant to all datasets. To illustrate them, we have chosen one very popular collection — INSPEC [13].

— The INSPEC gold standard contains keyphrases that may not occur in the texts. Some authors (e.g., [28, 23]) use the default (**long or full**) version of the gold standard, whereas others (e.g., [14, 34]) remove from the references keyphrases that do not occur in the texts (**short** version). The latter increases the F1 score.

— If a shortened gold standard is used, texts in which no gold-standard phrases appear may be removed from the dataset (e.g. in [20]). The final macro-average F1-score is higher in this way.

— Most studies present the results obtained on the subset of 500 documents from INSPEC-Test. However, some studies use 500 documents not only from INSPEC-Test but instead from the entire INSPEC dataset. For instance, the authors of [35] pick 500 documents out of 2000 texts where all the gold standard phrases occur in the selected documents. This must be accounted for, since comparing two algorithms is unreliable when their scores are computed on different text subsets rather than the same texts.

— INSPEC has two versions of the gold standard - uncontr and contr (details in [13]). Most studies employ the gold standard unconter set for evaluation, and some authors prefer to merge the conter and unconter sets as in [19]. In SemEval 2010, there are also multiple gold standards—keyphrases assigned by the authors and keyphrases assigned by readers to the texts. This can lead to confusion if it is not specified which set, or a combination thereof is used.

Of all the factors listed, the most significant impact comes from the first point that asks what is used as the gold standard: long or short variant.

We need to pay special attention when a short version of the gold standard is used and the original dataset texts have been shortened. For example, the authors of [22] use the short version of the gold standard and the SemEval 2010 collection. However, when extracting keyphrases, they do not use the full article texts from this dataset; instead, they rely only on the titles and abstracts. Only about 40% of the gold standard phrases appear in titles and abstracts. In this case, using the short version of the gold standard results in removing about 60% of the gold standard phrases. This substantially increases Recall and, consequently, the F1-score.

The F1 values obtained under this combination (titles and abstracts only + short gold standard) are comparable only to results produced in the same way. Whether such results can be meaningfully compared with those obtained on full-article texts requires clarification. In the short gold standard

built for full texts, a significantly larger proportion of the original keyphrases remains than 40%.

3.2.2 F1-score Calculation Issues

Let's highlight the following F1 calculation issues:

— Some works employ the micro-average F1 (e.g., [14]) for evaluation, and some studies use macro-average F1 (e.g., [31, 21]). Sometimes, authors report the F1-score without specifying whether the micro- or macro-average F1 was used. Still, the results obtained by each method differ and are incomparable.

— When calculating the macro-average F1-score, we also observe differences. For instance, in [3], the macro-average F1-score for the dataset is computed as the mean of all F1-scores calculated for each document in the dataset. In [20], where macro-averaged F1 is also declared, the authors first calculate macro-averaged Precision and Recall across all documents in the dataset. They then compute the final F1-score using these macro-averaged values in the F1 formula, as demonstrated by the available project code[1]. This calculation method yields results that differ from the standard macro-averaged F1 calculation.

— Lowercasing and stemming of extracted phrases and gold-standard phrases before calculating the F1-score is a widely accepted practice; however, it is not always stated in papers whether this step is performed.

3.2.3 Variations in Candidates Extraction Methods

We will examine this issue using algorithms that consist of two stages: candidate phrase extraction and candidate phrase ranking.

In the domain, it has not been common practice to evaluate separately the quality of the candidate set selected before ranking and the quality of the extracted keyphrases after ranking. Consequently, it remains unclear which stage contributes more to improving an algorithm's quality. Nevertheless, the candidate set obtained prior to ranking naturally influences the ranking and the overall performance of the algorithms.

One of the features that can enhance the set of candidate phrases is part-of-speech information and extraction of candidates as noun phrases. In the KE domain, it has been shown that most phrases in scientific texts are noun phrases, and nouns and adjectives are the main parts of speech used to extract keyphrases. Therefore, algorithms with PoS-tagging have an advantage over those that do not.

Original YAKE, RAKE, and KP-Miner algorithms are language-independent and do not require PoS-tagging tools, which is their advantage. However, it requires them to extract candidate phrases either as n-grams or as sequences of words between delimiters:

— In the case of n-grams, this results in many generated candidates, affecting both processing time and result quality during ranking in Precision and F1-score.

— When extracting sequences of words between delimiters, we obtain phrases that a less accurate than those extracted as noun phrases.

Thus, language-independent methods perform worse than methods with PoS-tagging during the candidate phrase extraction stage, and comparing these two types of algorithms is not entirely appropriate.

If we modify the candidate-phrase selection procedure in the language-independent algorithms YAKE, RAKE, and KP-Miner to use noun phrases, we observe improved performance. By extracting noun phrases in YAKE, RAKE, and KP-Miner in the same way as in language-dependent methods, we can directly compare the ranking approaches of these algorithms. This allows us to determine when algorithms perform better due to more effective candidate-phrase ranking and when they outperform others because their

[1] https://gitlab.com/matej.martinc/tnt_kid/-/blob/master/eval.py?ref_type=heads

candidate-extraction approach yields phrases with higher F1-scores before ranking.

The authors of [9] point out that, to achieve a fair and meaningful comparison, consistent and directly comparable experimental setups should be employed. In particular, they emphasize the importance of candidate-phrase extraction methods, since this choice determines both the upper bound on recall and the number of irrelevant candidates the algorithm will subsequently process. To address this, [9] standardised the candidate-extraction step for all algorithms in their study and extracted candidate phrases as noun phrases. We will follow this practice and do the same in our experiments.

3.2.4 Differences in Data Processing Tools and Stop-Word Lists

Let us show how stopwords can affect algorithm performance. In [35], the authors proposed a new KE algorithm and compared it with several others. One of these algorithms, from [16], was not outperformed. However, the authors of [35] argue that they, in fact, achieve better results than [16] — and they explain why:

— The authors of [16] extended the stopword list with words too common to be part of keyphrases. The performance of the algorithm from [16] drops by about 5% if these additional words are not included in the stopword list.

This 5% drop in the evaluation score allows [35] to outperform [16]. Therefore, [35] conclude that their algorithm is better than the one from [16], because the previous advantage of [16] was due to the extra stopwords, not the algorithm itself.

In our study [25], we demonstrated that using extended stopword lists leads to statistically significant improvements in the performance of unsupervised KE algorithms, on average by 4.5–6%.

These examples demonstrate how the use of extended stop-word lists can affect an algorithm's evaluation score. Even when only standard stop lists are used, we observe in [25] that different lists can be utilized in various studies. For example, the FOX stop list [8] is employed in [33], whereas the NLTK stop list is used in [3]. In most works, the authors do not specify information about the stop lists they have used.

Since stopwords are widely utilized in the domain, it is essential to specify which stop-word list was used and any modifications to this list. Without this clarification, comparisons between algorithms may be confounded. It makes it unclear whether one algorithm extracts better keyphrases than another or whether evaluation results differ because different stop-word lists were exploited.

Among other tools that can have an effect, but we suppose that this effect will be less than from extended stop-word lists, we will highlight the following: a way to process hyphens, instruments for tokenization, sentence splitting, and part-of-speech (POS) tagging.

3.2.5 Differences in Datasets

We highlight several points:

— Text length and position features.

— Single-word keyphrases in datasets with reader-assigned keyphrases.

Text length and position features. In scientific papers, most keyphrases are typically located at the beginning (title, headers, abstract, and introduction) and conclusion [18][24]. When processing full scientific articles, features as position that considers the index of the first phrase appearance in the text, text truncation, or the prioritization of phrases appearing in the title and abstract become particularly important. It occurs for the following reasons:

— Keyphrases are not evenly distributed throughout a scientific text [18][24]. They are most dense at the beginning. Therefore, if we extract candidates only from the start of the text (for example, from the title and abstract), we get a smaller candidate set that contains a higher proportion of true keyphrases.

— Algorithms that use position features usually give higher priority to phrases that appear near the beginning of the text. In practice, this works like extracting candidate phrases only from the start of the article, because phrases that appear later receive low ranks and are effectively ignored.

— Without positional features, extracting candidates from the full text produces a very large number of candidate phrases. All of them have the same priority — none are down-ranked or ignored. Only a small fraction of these phrases are true keyphrases. This makes ranking harder and lowers the overall evaluation score.

In [27], results are presented for the same algorithm run on the full SemEval2010 texts with and without the position feature. Not using the position feature leads to a significant decrease in the algorithm's performance, dropping in micro-average F1-score from 14.2 to 3.7

An illustration of the importance of the positional feature is presented in [4] (Section 6.3): the authors tested the features of the unsupervised YAKE algorithm. In this algorithm, the position feature is not based directly on the first occurrence of a phrase. Instead, the authors use the median position of the sentences in which the terms of the phrase appear. We expect that YAKE's positional feature will place less emphasis on phrases at the beginning of the text compared to the feature that weights phrases based on their first occurrence.

The materials in [4] are based on the following experiment. The authors merged the documents from all used datasets and split them into four quartiles of fixed length ([0, Q1]; [Q1, Q2]; [Q2, Q3]; [Q3, 24k]). For instance, [0, 153] indicates that 25% of all documents have a size between 0 and 153 terms, [153, 2k] between 153 and 2k terms, [2k, 5k] between 2k and 5k terms, and [5k, 24k] between 5k and 24k terms. They then tested the effectiveness of the YAKE method in each quartile.

The results showed that YAKE achieves its highest performance on texts in the second quartile (texts with size between [153, 2k]), performs slightly worse on documents in the first quartile, and its performance declines on documents in the third quartile, reaching a minimum in the fourth quartile. This example demonstrates that the quality of an algorithm's performance can deteriorate as text length increases strongly. Furthermore, it illustrates that longer texts can present greater challenges for KE algorithms compared to shorter texts. We suppose that length has an especially strong impact on algorithms with no or weak positional features.

We distinguish between strong and weak positional features based on how strongly they prioritize phrases appearing earlier in the text. For example, in [36, 24] the positional feature directly increases the weight of phrases with early first occur in the text: strong positional feature. In [4, 7], the position feature is present, but it is used indirectly and has a weaker influence on phrase weighting and ranking.

When considering the role of positional features, testing the unsupervised RAKE algorithm on full-text articles is of limited use [28]. Because RAKE does not use positional features, it performs worse on long texts. Therefore, when comparing other algorithms to RAKE, it is better to use datasets with short texts, such as abstracts. In the latter, RAKE shows better performance and evaluation scores. The same is true for other algorithms that do not use positional features, e.g. TextRank [23].

Authors can initially test algorithms on shortened versions of texts [20, 30, 5]. Also, text truncation is a common practice when KE algorithms use e.g. Bert [6], due to their limitations regarding text length. This must be taken into account. If evaluation results are obtained on truncated texts, such results cannot be fairly compared with those obtained on full texts. This is especially important when, in the latter case, the algorithm applied to full texts does not use a positional feature.

Keyphrase assignment and single-word phrase problem. Our study [26] shows that for datasets with reader-assigned keyphrases, there exists an issue with single-word phrases, a problem that is particularly pronounced in the INSPEC collection. The core of the problem lies in the observation that, on these datasets, algorithms that minimize the number of single-word phrases perform better. As shown in [26], removing single-word phrases from keyphrase candidates for such datasets improves the results in most

cases. Therefore, when evaluating the quality and comparing algorithms on these datasets, it is useful to assess how these algorithms perform, if they are required to extract only multi-word keyphrases. Otherwise, it becomes challenging to determine whether one algorithm outperforms another due to the involvement of new useful features or because the algorithm is designed to extract only a minimal number of single-word phrases.

3.3 Evaluation Pipeline 'Q-10': 10 Evaluation Questions

We proposed the 'Q-10' evaluation pipeline that considers the issues discussed above. This pipeline contains 10 questions that should be answered when evaluating KE algorithms. Of course, some factors may remain overlooked. Nevertheless, we aimed to gather most of the issues related to KE algorithm evaluation and to explain their importance. To the best of our knowledge, no similar studies have been conducted in the domain.

Processing specific questions/parameters:

— **Q-1**: What is the dataset's gold standard version? Does each text in the dataset have a long (all assigned keyphrases) or short (only assigned keyphrases that occur in the text) gold standard?

— **Q-2**: If a micro- or macro-average F1 score was used? How was the score calculated? For example, source code provided for articles [3][2] and [20][3] shows that the calculation of the macro-average F1-score can be realized in different maner. Additional question when a short gold standard is used: Have texts been removed from the dataset if their gold standard does not contain any phrases that occur in the text?

— **Q-3**: Were the extracted phrases and gold-standard phrases stemmed and lowercased before evaluation?

— **Q-4**: Which keyphrase candidate extraction method was used, and was the same method applied to all other algorithms in the comparison? For datasets with reader-assigned keyphrases, how were single-word keyphrase candidates processed?

— **Q-5**: What stop-word list and its modifications were used?

— **Q-6**: If the algorithm's performance is evaluated on full texts, does the algorithm incorporate a positional feature? Additional questions: How are hyphens processed in keyphrase extraction, and which tools are used for tokenization, sentence splitting, part-of-speech (POS) tagging, and other related preprocessing steps?

Dataset specific questions:

— **Q-7**: What texts are used as a test set for each specific dataset?

— **Q-8**: If the gold standard includes multiple variants (e.g., keyphrases assigned by readers and keyphrases assigned by authors), which keyphrases or combination of keyphrases are used?

— **Q-9**: Does the dataset contain short (title+abstract) or long texts (scientific articles)?

— **Q-10**: Were keyphrases for the texts assigned by readers or by authors?

3.4 Comparison of KE algorithms: cCoss-paper, Reimplementation, PKE

The main issue with cross-paper algorithm comparisons is the uncertainty about which results are truly comparable. Authors often provide detailed descriptions of their proposed algorithms but offer only limited evaluation details. In some cases, the entire evaluation process is reduced to a single statement that the F1-score was used, without specifying whether micro or macro averaging was applied. Information about which version of the gold standard (long or short) was used is frequently missing. Details about stemming are sometimes left out. Sometimes it is not

[2] https://github.com/boudinfl/pke/blob/master/examples/benchmarking-models.ipynb

[3] https://gitlab.com/matej.martinc/tnt_kid/-/blob/master/eval.py?ref_type=heads

mentioned that extra words were added to the standard stopword lists. Clearly, the lack of detailed information about the evaluation process makes meaningful comparisons between studies difficult. The 'Q-10' evaluation pipeline questions allow us to determine which evaluation scores can be cross-paper compared and which cannot.

If the algorithms being compared are re-implemented by authors, it is assumed that the evaluation scores are calculated identically for all algorithms. However, such an approach in keyphrase extraction has its limitations.

A good example is the paper [11], where the authors compare results from the original papers with results from their own re-implementations. Because the results in the original papers and in [11] are different, the authors carefully re-checked their re-implementation and concluded that it did not contain any errors. When analyzing this difference, the authors of [11] reported, "we speculate that document pre-processing (e.g., stemming) has contributed to the discrepancy, but additional experiments are needed to determine the reason."

Independent reimplementation of KE algorithms for comparative studies is challenging because implementation errors must be excluded. The only way to check this is by reproducing original results, which can be difficult or even impossible [11], and full equivalence between independently developed implementations cannot be guaranteed. A practical solution is therefore to rely on existing, validated reimplementations instead of creating new ones.

There are commonly used frameworks in the domain that contain reimplementations of KE algorithms. One of the most famous is PKE [3]. The use of this tool helps ensure that standardized implementations of the algorithms are employed and that potential errors arising from reimplementing are eliminated. At the same time, PKE is a flexible tool, allowing authors to use their own approaches to evaluation score calculation, their own stopword lists, both full and short versions of gold standards, etc. While this tool removes differences in algorithm implementations, it does not resolve the issue of varying methods for computing quality scores.

We assume that combining PKE with the proposed 'Q-10' evaluation pipeline will allow a standardized and reliable evaluation process. PKE provides source code on the tool's page[4] where the preprocessing and quality evaluation steps are realized identically as they are described in the original PKE paper [3]. All results in [3] are fully reproducible with this code, which transparently details the preprocessing and evaluation process. We exploited this code in our experiments below.

4 Experiment Description

We evaluated ten unsupervised algorithms on six datasets using different methods of macro-average F1-score calculation. The obtained results were collected. We carried out these experiments to achieve the following:

— To show how F1-scores change when different F1 calculation methods are applied to the same algorithms and datasets. We calculate the percentage change in F1-score values for these cases. The latter demonstrates how evaluation results vary when answers to the questions in the 'Q-10' evaluation pipeline differ.

— Establish a comparative baseline for unsupervised KE algorithms. If cross-paper F1 evaluation with unsupervised KE is used, researchers can choose the most suitable scores depending on which F1-score calculation method they use.

We collected results for supervised algorithms from articles whose F1-score calculation methods are the same as those we use. Results for *KEA, *CorrRNN, and *CopyRNN are taken from [9]. The results for the **GPT-2+BiLSTM-CRF and **TNT-KID algorithms are taken from [20]. This allowed us:

— Based on the 'Q-10' evaluation pipeline, to perform a cross-paper comparison of results between unsupervised KE algorithms and RNN- and transformer-based algorithms.

[4] https://github.com/boudinfl/pke

4.1 DataSets

We divided all datasets into two groups [26]:

— Datasets with reader-assigned keyphrases (INSPEC [13], SemEval 2017 [1]) or reader- and author-assigned keyphrases (SemEval 2010 [15]);

— Datasets with author-assigned keyphrases (KP20k [22], KPBiomed [12], PubMed [29]).

All datasets consist of scientific texts. The INSPEC, KP20k, KPBiomed, and SemEval2017 datasets contain short texts: the first three consist of titles with abstracts, while the last comprises selected paragraphs.

The SemEval2010 and PubMed datasets include full articles. We used only the titles and abstracts from SemEval2010 text. For PubMed we truncated each text from the 1200th character onward [26]. Thus, in all experiments below we used short texts. This allowed us to avoid considering the presence or absence of positional features in the evaluated algorithms.

As a test collection in PubMed, we used the first 500 documents. In KPBiomed, we used the first 2,000 documents from the KPBiomed test set. In both cases, texts were selected using Python's islice function. The datasets were loaded into PKE from the repository[5].

4.2 Keyphrase Extraction Methods

We utilized ten unsupervised methods and their re-implementations available in PKE [3]: FirstPhrases, TextRank, SingleRank, TopicRank, MultipartiteRank, PositionRank, TopicalPageRank, TfIDF, KPMiner, and YAKE. All these algorithms were employed in a two-stage process: first, candidate phrases were extracted, and then the candidate phrases were ranked. The top k-ranked phrases were selected as keyphrases.

Candidate phrases were extracted as noun phrases in one of two ways:

— **pattern P1**:
$< ADJ > * < NOUN|PROPN > +$.

— **pattern P2**:
$NBAR:$
$< NOUN|PROPN|ADJ > * < NOUN|PROPN >$
$NP:$
$< NBAR >< NBAR >< ADP >< NBAR >.$

Candidate phrases were ranked according to the specific ranking method of each algorithm. Then the top k-ranked candidates were extracted as keyphrases.

All text pre-processing, dataset loading, and related procedures were performed as described in PKE project[6].

4.3 F1-score Calculation Methods

We replicated the evaluation methods exploited macro-average F1-score described in several articles. The description of the F1-score calculation methods in accordance with the 'Q-10' pipeline is presented in Table 1 and Table 2.

— Method **F**: we replicate the approach described in [9] and follow the same evaluation process as in PKE [3]. In this case, the long version of the gold standard is used to calculate Recall. The final F1-score is then obtained as the average F1-score across all texts.

— Method **F+**: it is method F with the following modifications. A short version of the gold standard is used. Following [20], we remove texts from the dataset if their gold standard does not contain any phrases occurring in the text. The final F1-score is then obtained as the average F1-score across all texts.

— Method **S**: replicates the approach described in [20]. A short version of the gold standard is used. Texts are removed from the dataset if their gold standard does not contain any phrases that occur in the text. The final F1-score is calculated using the F1 formula, where Recall is the macro-averaged Recall over all texts in the dataset, and Precision is the macro-averaged Precision over all texts in the dataset.

[5] https://huggingface.co/taln-ls2n

[6] https://github.com/boudinfl/pke/blob/master/examples/benchmarking-models.ipynb

— Methods **S-+** and **S++**. Methods S-+ and S++ differ from S in data processing and algorithm parameters. Specifically, in S-+ and S++, an extended stop words list (ExtendedSW [25]) and an improved phrase extraction pattern (pattern P2 instead P1) are used. In S++, single-word phrases are filtered out from the set of candidate phrases. It is done for datasets with reader-assigned keyphrases. For datasets where keyphrases were assigned solely by the author, one-word phrases were not removed and we use S-+ for this case. Since SemEval 2010 is most often used with the gold standard that combines keyphrases assigned by readers and authors, we used the same gold standard. Therefore, for this dataset, we report both S-+ and S++ values.

When evaluating the obtained keyphrases with the described methods, the F1-score is denoted **F1@k**, where k is the maximum number of keyphrases that can be extracted for each text:

— In datasets where phrases are assigned by readers, the number of phrases in the gold standard is around ten or more [26]; for these datasets, we used F1@10.

— For datasets with author-assigned keyphrases, the number of phrases in the gold standard is, on average, close to five [26]; therefore, we used F1@5.

— For the PubMed collection, both F1@5 and F1@10 results are included for comparison purposes.

Extracted and gold standard phrases were stemmed and lowercase before comparison.

5 Results and Discussion

5.1 Description of the Tables

The results are presented in Table 3 for datasets with reader-assigned keyphrases and in Table 4 for datasets with author-assigned keyphrases:

— Evaluation results for unsupervised KE algorithms obtained in this paper: the abbreviated names F, F+, S, S++, and S-+ correspond to the F1 calculation methods described in Section 4.3.

— Cross-paper evaluation results for supervised algorithms marked with * are taken from [9] and are used only for comparison with F1-score results calculated in the same manner as in [9] (method F).

— Cross-paper evaluation results for supervised algorithms marked with ** are taken from [20] and are used only for comparison with F1-score results calculated in the same way as in [20] (method S).

The following data are also presented in the tables:

— **F**or unsupervised KE algorithms: symmetric percent difference is presented between the F1@10/F1@5 scores calculated as in [9] and as in [20]".

— For unsupervised KE algorithms: the average symmetric percent difference in F1@10/F1@5 is provided when these scores are calculated as described in method S and in methods S++/S-+.

— Improvement achived buy best supervised methods over best unsupervised in F1@10/F1@5".

5.2 Results and Discussion

Table 3 and Table 4 show the following:

— Differences in evaluation scores are substantial when the same algorithms are evaluated with F1 on the same datasets using different F1 calculation methods: when calculated as in [9] versus as in [20]. Across datasets, these differences range from 7% to 36%. and must be taken into account when comparing algorithms. Primarily, this discrepancy depends on the percentage of gold-standard phrases that do not occur in

Table 1. ”Q-10” pipeline: F1-score calculation method, low.-lowercasing, Cand.extract.-candidate extraction method

F1 calculation $->$	F as in [9]	F+	S as in [20]	S++	S-+
Q1: GD version	full	short	short	short	short
Q2: F1-score calculation: macro-avg and other details	F1 - mean of F1 for texts. No texts are removed from the dataset	F1 - mean of F1 for texts. If no phrases from the gold standard occur in a text, that text is removed from the dataset	macro-avg Precision and Recall are used to calculate final F1-score. If no phrases from the gold standard occur in a text, that text is removed from the dataset	macro-avg Precision and Recall are used to calculate final F1-score. If no phrases from the gold standard occur in a text, that text is removed from the dataset	macro-avg Precision and Recall are used to calculate final F1-score. If no phrases from the gold standard occur in a text, that text is removed from the dataset
Q3: Stemming+low.	yes	yes	yes	yes	yes
Q4: Cand.extract.	Pattern P1	Pattern P1. Single-word phrases are removed from the keyphrase candidates	Pattern P1	Pattern P2. Single-word phrases are removed from the keyphrase candidates	Pattern P2
Q5: Stop-words list	NLTK	NLTK	NLTK	SWExtended	SWExtemded.
Q6: Positional feature for long texts, NLP tools	- (short text), NLP tool: built-in PKE	- (short text), NLP tool: built-in PKE	- (short text), NLP tool: built-in PKE	- (short text), NLP tool: built-in PKE	- (short text), NLP tool: built-in PKE

Table 2. ”Q-10” pipeline: DataSet. SE2017-SemEval2017, SE2010-SemEval2010, Assign.(keyphrases assigned by)

DataSet $->$	INSPEC	SE2017	SE2010	KP20k	PubMed	KPBiomed
Q7: Test set	Test	Test	Test	Test	first 500 doc.	first 2,000 doc (test)
Q8: GD keyphr.	Uncontr	N/A	Author & Reader	N/A	N/A	N/A
Q9: Text Length	short: title & abstract	short: paragraphers	short from long: take title & abstract only	short: title & abstract	short from long: texts' first 1,200ch.	short: title & abstract
Q10: Assigem.	Reader	Reader	Author & Reader	Author	Author	Author

Table 3. Results: dataSets with reader assigned keyphrases

INSPEC				**SemEval2017**				**SemEval2010**				
Reader				Reader				Reader+Author				
F [9]	F+	S [20]	S++	F [9]	F+	S [20]	S++	F [9]	F+	S [20]	S++	S-+
Unsupervised approaches F1@10												
FirstPhrases												
30.1	31.3	33.4	49.0	19.8	20.2	21.2	28.7	16.6	22.7	24.1	27.9	26.2
TextRank												
37.3	38.5	40.9	47.2	23.5	24.1	25.3	27.8	15.8	20.4	21.6	24.3	24.7
SingleRank												
36.5	38.4	40.9	47.8	25.9	26.5	27.9	28.8	18.4	25.7	27.4	27.7	29.2
TopicRank												
29.6	31.4	33.4	47.3	20.6	21.1	22.2	25.1	15.1	21.7	23.0	23.8	23.5
MultipartiteRank												
30.5	32.4	34.5	48.7	21.7	22.1	23.4	27.7	16.1	22.6	24.0	25.5	25.1
PositionRank												
34.8	36.9	39.3	47.7	25.7	26.3	27.6	28.8	18.2	26.1	27.7	28.6	29.2
TopicalPageRank												
36.3	38.1	40.5	47.8	**26.2**	26.8	28.2	29.1	18.1	24.8	26.5	27.6	28.4
TfIdf												
36.3	39.2	**41.7**	48.8	24.9	25.5	26.9	28.3	16.8	24.1	25.6	26.3	26.6
KPMiner												
35.1	37.2	39.6	48.8	25.0	25.6	27.0	28.6	17.9	25.5	27.0	28.5	28.0
YAKE												
36.0	38.7	41.2	48.2	24.7	25.2	26.6	**28.8**	**18.6**	27.6	**29.2**	28.5	29.4
Supervised approaches F1@10												
*KEA												
34.5	N/A			-				19.5	N/A			
*CorrRNN												
27.9	N/A			-				19.4	N/A			
*CopyRNN												
28.2	N/A			-				**20.3**	N/A			
**GPT-2+BiLSTM-CRF												
N/A		52.4	N/A	-				N/A		27.4	N/A	
**TNT-KID												
N/A		**53.6**	N/A	-				N/A		**33.7**	N/A	
Differences in evaluation scores												
Symmetric percent difference between the F1@10 calculation methods [9] and [20]												
12%				7%				36%				
Symmetric percent difference between the F1@10 calculation methods S and S++												
22%				10%				11%				
% Improvement of best supervised methods over best unsupervised in F1@10												
-8%	-	29%	-	-				9%	-	15%	-	-

Table 4. Results: dataSets with author assigned keyphrases

kp20k				**KPBiomed**				**PubMed**				**Pubmed**
Author				Author				Author				Author
F [9]	F+	S [20]	S-+	F [9]	F+	S [20]	S-+	F [9]	F+	S [20]	S-+	F
Unsupervised approaches F1@5												**F1@10**
FirstPhrases												
15.1	18.9	20.3	21.2	16.4	19.6	20.3	**21.3**	15.7	18.7	19.7	20.4	15.4
TextRank												
7.3	9.2	9.8	10.7	4.8	5.8	6.0	6.4	4.4	5.4	5.6	6.6	8.2
SingleRank												
10.8	13.7	14.6	15.4	8.3	10.1	10.5	10.9	8.7	10.5	10.9	11.9	12.4
TopicRank												
13.1	16.4	17.7	18.2	13.7	16.4	17.1	17.5	13.8	16.4	17.1	17.9	14.0
MultipartiteRank												
14.5	18.1	19.6	20.0	15.4	18.5	19.2	19.7	15.6	18.5	19.3	19.6	16.1
PositionRank												
13.2	16.7	17.9	18.6	11.7	14.1	14.6	15.1	12.4	14.8	15.5	16.4	15.2
TopicalPageRank												
11.3	14.2	15.2	16.0	8.6	10.5	10.9	11.5	9.6	11.6	12.0	12.9	12.7
TfIdf												
13.7	17.2	18.3	18.6	16.5	19.7	20.5	20.8	18.3	21.8	22.8	23.1	16.4
KPMiner												
15.3	19.2	**20.4**	20.8	**16.8**	20.1	21.0	21.2	**18.5**	21.9	22.9	**23.2**	16.6
YAKE												
14.8	18.7	19.9	20.4	14.8	17.7	18.4	19.0	13.9	16.6	17.5	18.4	**16.6**
Supervised approaches F1@5												**F1@10**
*KEA												
N/A				-				N/A				18.6
*CopyCNN												
N/A				-				N/A				**24.2**
*CorrCNN												
N/A				-				N/A				20.8
**GPT-2 + BiLSTM-CRF												
N/A		**35.5**	N/A	-				-				-
**TNT-KID												
N/A		33.6	N/A	-				-				-
Differences in evaluation scores												
Symmetric percent difference between the F1@5 calculation methods [9] and [20]												
29%				22%				22%				-
Symmetric percent difference between the F1@5 calculation methods S and S-+												
4%				3%				6%				-
% Improvement of best supervised over best unsupervised in F1@5												in F1@10
-	-	74%	-	-				-				46%

the text. The results show that comparing evaluation scores obtained using different versions of the gold standard (long/short) is not acceptable.

— The average difference in evaluation scores obtained using method S [20] and method S++ is as follows. For datasets where keyphrases are *assigned by readers*, S++ gives a clear improvement. The difference ranges from 10% to 22%. The effect of removing single-word phrases from the candidate set is especially strong here, with the largest impact on INSPEC. For datasets with *author-assigned* keyphrases, the average percentage difference between the S and S-+ evaluation scores is smaller. It ranges from 3% to 6%. The difference is mainly caused by using different stopword lists.

— The difference in evaluation scores between RNN- or transformer-based and unsupervised KE algorithms varies significantly across datasets where keyphrases are assigned by readers versus by authors. In datasets with author-assigned keyphrases, RNN- and transformer-based methods outperform unsupervised approaches by 46% and 74%, respectively. In datasets with reader-assigned keyphrases, the performance gap between RNN-/transformer-based approaches and unsupervised KE methods ranges from -8% on INSPEC to 15% on SemEval 2010.

5.3 Cross-Paper Evaluation Checking

In this paper, we implemented F1 evaluation methods similar to those described in [9] (method F) and [20] (method S). Therefore, wherever possible, we compared our unsupervised KE results with the results reported in [9] and [20]. The goal was to verify whether the results we obtained would coincide with those reported in other publications for the same algorithms applied to the same datasets and under identical settings of the 'Q-10' pipeline.

From [9], the performance of five algorithms on two datasets is directly comparable with our results. This comparison is shown in Table 5, using F1@10 for the INSPEC and KP20k datasets. Both datasets contain only titles and abstracts. The F1-score values almost completely coincide and show only minimal differences. The only exception is TextRank, where the discrepancy is noticeable: 35.8 vs 37.3. We assume that this difference may be related to possible variations in word-weight estimation based on the PageRank-like formula used in the algorithm.

We also compared our results on the SemEval2010 and PubMed datasets with the results reported in [9] for the same algorithms. In our experiments, we used only the titles and abstracts, while [9] used the full articles from these datasets.

This comparison shows whether algorithms can perform better when keyphrases are extracted from titles and abstracts instead of from full texts. The results for these datasets are also shown in Table 5.

These results show that FirstPhrase, TextRank, PositionRank, and MPRank achieve higher scores when only titles and abstracts are used, instead of full article texts. This further supports the importance of the position feature and the choice of abstracts for the experiments in this study.

Comparison with the results from [20] is not possible. The authors of [20] report two types of results: (1) results taken from related work, and (2) results obtained by their own re-implementations of the algorithms. We do not use the results taken in [20] from related work, because too many evaluation details are unknown. Among the re-implemented algorithms, the only one that overlaps with our study is YAKE. However, we assume that the candidate-selection steps in [20], as in the original YAKE, differ from those in our work, where we use noun phrases as candidates. This makes the results not directly comparable.

6 Conclusion

This research aims to bridge the gap in the evaluation process and demonstrate how significantly the macro-average F1-score can vary when computed for the same algorithms and datasets in different ways. We showed that this discrepancy can range from 7% to 36% across datasets. Such a large difference in F1 values

Table 5. Comparing: results were obtained, and results are presented in [9]. Method F was used to calculate macro-average F1@10, abs.- titles+abstractes were processed, full - full papers were processed

F@10	INSPEC		kp20k		SemEval		PubMed	
	[9] abs.	our abs.	[9] abs.	our abs.	[9] full	our abs.	[18] full	our abs.
FirstPhrase	29.3	30.1	13.5	13.5	13.8	16.6	15.4	15.5
TextRank	35.8	37.3	10.2	10.1	3.5	15.8	1.8	8.2
TfIdf	36.5	36.3	11.6	12.1	17.7	16.8	16.7	16.4
PositionRank	34.2	34.8	14.1	14.1	6.8	18.2	4.9	15.2
MPRank	30.5	30.5	13.6	13.6	14.3	16.1	15.8	16.1

for the same algorithms and datasets is not acceptable. Therefore, it is necessary to examine and take into account the specific details of how F1 is calculated that lead to these large differences. In this paper, we show that this problem exists. As a way to address it, we propose the 'Q-10' pipeline, which defines ten key points that must be considered when calculating F1-scores for comparing KE algorithms.

Additionally, using the 'Q-10' evaluation pipeline, we conducted a cross-paper comparison of the results for unsupervised and RNNs-/transformer-based KE approaches.

Also, we collected quality evaluation scores for unsupervised KE algorithms using different F1-score calculation approaches. We aimed to create a set of baselines, from which evaluation scores can be selected depending on the chosen F1-score calculation method.

References

1. **Augenstein, I., Das, M., Riedel, S., Vikraman, L., McCallum, A. (2017).** SemEval 2017 task 10: ScienceIE - extracting keyphrases and relations from scientific publications. **Bethard, S., Carpuat, M., Apidianaki, M., Mohammad, S. M., Cer, D., Jurgens, D.**, editors, Proceedings of the 11th International Workshop on Semantic Evaluation (SemEval-2017), Association for Computational Linguistics, Vancouver, Canada, pp. 546–555.
2. **Bernardini, A., Carpineto, C., D'Amico, M. (2009).** Full-subtopic retrieval with keyphrase-based search results clustering. IEEE/WIC/ACM International Joint Conference on Web Intelligence and Intelligent Agent Technology, IEEE, pp. 206–213.
3. **Boudin, F. (2016).** pke: an open source python-based keyphrase extraction toolkit. Proceedings of COLING 2016, the 26th International Conference on Computational Linguistics: System Demonstrations, Osaka, Japan, pp. 69–73.
4. **Campos, R., Mangaravite, V., Pasquali, A., Jorge, A., Nunes, C., Jatowt, A. (2020).** Yake! keyword extraction from single documents using multiple local features. Information Sciences, Vol. 509, pp. 257–289.
5. **Choi, M., Gwak, C., Kim, S., Kim, S., Choo, J. (2023).** SimCKP: Simple contrastive learning of keyphrase representations. **Bouamor, H., Pino, J., Bali, K.**, editors, Findings of the Association for Computational Linguistics: EMNLP 2023, Association for Computational Linguistics, Singapore, pp. 3003–3015.
6. **Devlin, J., Chang, M., Lee, K., Toutanova, K. (2019).** BERT: pre-training of deep bidirectional transformers for language understanding. **Burstein, J., Doran, C., Solorio, T.**, editors, Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, NAACL-HLT 2019, Minneapolis, MN, USA, June 2-7, 2019, Volume 1 (Long and Short Papers), Association for Computational Linguistics, pp. 4171–4186.
7. **Florescu, C., Caragea, C. (2017).** Positionrank: An unsupervised approach to keyphrase extraction from scholarly documents. Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics, volume 1, Association for Computational Linguistics, pp. 1105–1115.
8. **Fox, C. J. (1990).** A stop list for general text. SIGIR Forum, Vol. 24, No. 1-2, pp. 19–35.
9. **Gallina, Y., Boudin, F., Daille, B. (2020).** Large-scale evaluation of keyphrase extraction

models. Proceedings of the ACM/IEEE Joint Conference on Digital Libraries in 2020, JCDL '20, Association for Computing Machinery, New York, NY, USA, pp. 271–278.

10. **Gutwin, C., Paynter, G., Witten, I., Nevill-Manning, C., Frank, E. (1999).** Improving browsing in digital libraries with keyphrase indexes. Decision Support Systems, Vol. 27, No. 1–2, pp. 81–104.
11. **Hasan, K. S., Ng, V. (2010).** Conundrums in unsupervised keyphrase extraction: Making sense of the state-of-the-art. **Huang, C.-R., Jurafsky, D.**, editors, Coling 2010: Posters, Coling 2010 Organizing Committee, Beijing, China, pp. 365–373.
12. **Houbre, M., Boudin, F., Daille, B. (2022).** A large-scale dataset for biomedical keyphrase generation. **Lavelli, A., Holderness, E., Jimeno Yepes, A., Minard, A.-L., Pustejovsky, J., Rinaldi, F.**, editors, Proceedings of the 13th International Workshop on Health Text Mining and Information Analysis (LOUHI), Association for Computational Linguistics, Abu Dhabi, United Arab Emirates (Hybrid), pp. 47–53.
13. **Hulth, A. (2003).** Improved automatic keyword extraction given more linguistic knowledge. Proceedings of EMNLP'03, ACL, pp. 10–20.
14. **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, USA, pp. 216–223.
15. **Kim, S. N., Medelyan, O., Kan, M.-Y., Baldwin, T. (2010).** SemEval-2010 task 5 : Automatic keyphrase extraction from scientific articles. **Erk, K., Strapparava, C.**, editors, Proceedings of the 5th International Workshop on Semantic Evaluation, Association for Computational Linguistics, Uppsala, Sweden, pp. 21–26.
16. **Liu, Z., Huang, W., Zheng, Y., Sun, M. (2010).** Automatic keyphrase extraction via topic decomposition. Proceedings of the Conference on Empirical Methods in Natural Language Processing, Association for Computational Linguistics, pp. 366–376.
17. **Liu, Z., Li, P., Zheng, Y., Sun, M. (2009).** Clustering to find exemplar terms for keyphrase extraction. Proceedings of the Conference on Empirical Methods in Natural Language Processing, Association for Computational Linguistics, pp. 257–266.
18. **Lopez, P., Romary, L. (2010).** HUMB: Automatic key term extraction from scientific articles in GROBID. **Erk, K., Strapparava, C.**, editors, Proceedings of the 5th International Workshop on Semantic Evaluation, Association for Computational Linguistics, Uppsala, Sweden, pp. 248–251.
19. **Mahata, D., Kuriakose, J., Shah, R. R., Zimmermann, R. (2018).** Key2Vec: Automatic ranked keyphrase extraction from scientific articles using phrase embeddings. **Walker, M., Ji, H., Stent, A.**, editors, Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 2 (Short Papers), Association for Computational Linguistics, New Orleans, Louisiana, pp. 634–639.
20. **Martinc, M., Škrlj, B., Pollak, S. (2022).** Tnt-kid: Transformer-based neural tagger for keyword identification. Natural Language Engineering, Vol. 28, No. 4, pp. 409–448.
21. **Medelyan, O., Frank, E., Witten, I. H. (2009).** Human-competitive tagging using automatic keyphrase extraction. **Koehn, P., Mihalcea, R.**, editors, Proceedings of the 2009 Conference on Empirical Methods in Natural Language Processing, Association for Computational Linguistics, Singapore, pp. 1318–1327.
22. **Meng, R., Zhao, S., Han, S., He, D., Brusilovsky, P., Chi, Y. (2017).** Deep keyphrase generation. **Barzilay, R., Kan, M.-Y.**, editors, Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers), Association for Computational Linguistics, Vancouver, Canada, pp. 582–592.
23. **Mihalcea, R., Tarau, P. (2004).** TextRank: Bringing order into texts.
24. **Nguyen, T. D., Luong, M.-T. (2010).** WINGNUS: Keyphrase extraction utilizing document logical structure. **Erk, K., Strapparava, C.**, editors, Proceedings of the 5th International Workshop on Semantic Evaluation, Association for Computational Linguistics, Uppsala, Sweden, pp. 166–169.
25. **Popova, S., Alexandrov, M., Cardiff, J. (2024).** Stop-word lists in keyphrase extraction: Their influence and comparison. volume 28, pp. 1449–1459.

26. **Popova, S., Danilova, V., Alexandrov, M., Cardiff, J. (2024).** Unsupervised keyphrase extraction: Ranking step and single-word phrase problem. volume 28, pp. 1377–1391.
27. **Popova, S., Danilova, V., Cardiff, J. (2024).** Rapid unsupervised keyphrase extraction from single document. Proceedings of the XXth Conference of Open Innovations Association FRUCT, volume 36, pp. 248–251.
28. **Rose, S., Engel, D., Cramer, N., Cowley, W. (2010).** Automatic keyword extraction from individual documents. Text Mining, pp. 1–20.
29. **Schutz, A. (2008).** Keyphrase extraction from single documents in the open domain exploiting linguistic and statistical methods. Master's thesis, National University of Ireland.
30. **Song, M., Liu, H., Jing, L. (2023).** HyperRank: Hyperbolic ranking model for unsupervised keyphrase extraction. **Bouamor, H., Pino, J., Bali, K.**, editors, Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing, Association for Computational Linguistics, Singapore, pp. 16070–16080.
31. **Tsatsaronis, G., Varlamis, I., Nørvåg, K. (2010).** SemanticRank: Ranking keywords and sentences using semantic graphs. **Huang, C.-R., Jurafsky, D.**, editors, Proceedings of the 23rd International Conference on Computational Linguistics (Coling 2010), Coling 2010 Organizing Committee, Beijing, China, pp. 1074–1082.
32. **Turney, P. D. (2000).** Learning algorithms for keyphrase extraction. Information Retrieval, Vol. 2, No. 4, pp. 303–336.
33. **Villmow, J., Wrzalik, M., Krechel, D. (2018).** Automatic keyphrase extraction using recurrent neural networks. Machine Learning and Data Mining in Pattern Recognition: 14th International Conference, MLDM 2018, New York, NY, USA, July 15-19, 2018, Proceedings, Part II, Springer-Verlag, Berlin, Heidelberg, pp. 210–217.
34. **Wan, X., Xiao, J. (2008).** Single document keyphrase extraction using neighborhood knowledge. Proceedings of the 23rd National Conference on Artificial Intelligence - Volume 2, AAAI'08, AAAI Press, pp. 855–860.
35. **Wang, R., Liu, W., Mcdonald, C. (2015).** Using word embeddings to enhance keyword identification for scientific publications. pp. 257–268.
36. **Yu, Z., Johnson, T. R., Kavuluru, R. (2013).** Phrase based topic modeling for semantic information processing in biomedicine. 2013 12th International Conference on Machine Learning and Applications, volume 1, pp. 440–445.
37. **Zeng, H. J., He, Q. C., Chen, Z., Ma, W. Y., Ma, J. (2004).** Learning to cluster web search results, pp. 210–217.

Article received on 08/07/2025; accepted on 24/09/2025.
**Corresponding author is Svetlana Popova.*