

Intelligent Text Summarization Using a Fuzzy-Driven BART Architecture

Bharti Sharma¹, Minakshi Tomer¹, Charu Gupta^{2,*}, Tripti Rathee¹, Oscar Castillo³

¹ Maharaja Surajmal Institute of Technology,
India

² Bhagwan Parshuram Institute of Technology,
India

³ TecNM/Tijuana Institute of Technology,
Mexico

{bhartisharma, minakshi.tomar, rathee.tripti}@msit.in,
charu.wa1987@gmail.com, ocastillo@tectijuana.mx

Abstract. Every day there is huge amount of data on the internet on a specific subject, leading to the availability of massive amounts of data, which includes relevant and irrelevant search. Therefore, it is necessary to summarize the text data in order to understand its key idea in limited time. This research proposes a new hybrid text summarization framework utilizing fuzzy logic and BART model. In order to improve the quality, the proposed framework is used in conjunction with extractive and abstractive summarization methods. In extractive method fuzzy logic utilizes fuzzy rules to create sentence embedding and for abstractive summarization method BART utilizes sequence-to-sequence method in which the encoder is bidirectional BERT and the decoder is autoregressive (GPT-like). To generate an intermediate summary each sentence is scored based on relevance and novelty metric. The sentences with the highest scores are selected to be the part of summary, which is used as input to abstractive method. In order to evaluate the efficiency of the proposed model, extensive experiments on benchmark datasets are performed- for *abstractive summarization* CNN/Daily Mail, DUC-2004 and *extractive summarization* DUC-2006 and DUC-2007 are used for conducting the experiments and recall oriented understudy for Gisting Evaluation (ROUGE) is used for evaluation. The results indicates that the proposed model is capable of performing efficient text summarization and generate better results than other state-of-art models.

Keywords. Autoregressive, BART, fuzzy logic, ROUGE, text summarization.

1 Introduction

Text summarization has emerged as a powerful technique to quickly and accurately summarize a large amount of text [1]. It is important to create natural language summaries that are informative and accurate and can keep the main ideas of the source articles [2] Text summarization aims to deliver the most relevant information in a condensed form while preserving the essential meaning of the primary text [3]. Choosing the appropriate information and determining what is significant while mechanically summarizing text is the most difficult step. Automatic summarization has its own problems compared to other Natural Language Processing (NLP) tasks [4]. For example, input and output sequence lengths are often same in machine translation tasks, but input and output sequence lengths are often very different in summarization tasks. In addition, machine translation jobs typically need direct alignment of input and output sequences at the level of individual words, whereas this is less apparent in summarization. There are two types of techniques for automatically summarizing text: extraction and abstraction [5]. While abstractive summarization creates summaries based on the major ideas of the material, so the summaries could be rephrased in more general terms, extractive summarization creates summaries by taking significant portions of the source document

and putting them together verbatim [5]. As a form of summary that is close to how humans do it, abstractive approaches should be able to rewrite the source document's main concepts in a style that is grammatically correct and easily understood by humans.

Deep neural network models are being used a lot for NLP tasks. Majority of NLP tasks uses the attention-based sequence-to-sequence framework with recurrent neural networks (RNNs) [6]. But it is found that RNN-based models are more likely to have gradient vanishing because their non-linearity is set up in a chain, while CNN-based models are set up in a hierarchy [7]. Also, RNNs' training is inefficient since parallelization over sequence elements is impossible because of temporal dependencies among hidden states.

In this paper, Fuzzy Logic and BART model (FL-BART), a hybrid framework that combines extractive and abstractive approaches is proposed. Fuzzy [8] is used for the extractor, and BART [9-12] is used for the abstractor. The primary objective is to develop a framework that integrates extractive and abstractive summarization to address their respective shortcomings. The extractive phase picks out the important sentences, lowering dataset size and redundancy in preparation for the abstractive phase. The sentences from the extractive summarization are rephrased using newly generated words in the abstractive phase, making the resulting summary sound more like it was written by a person. By giving shorter, higher-quality input sentences, training times can be shortened by employing extractive summaries. In the proposed framework's extractive phase, Fuzzy Rules for features extraction trained on the CNN/Daily mail dataset [13] are applied. Fuzzy logic is applied to vector representation of data in order to produce more accurate semantic connections between sentences. A fuzzy rule can easily deal with long-term dependencies and varying-length sequential data. The BART model's next phase receives the previous phase's output as input. CNN/Daily Mail and DUC-2004 are used to train and test the hybrid framework. ROUGE [14] is used to evaluate and validate the method. The results of experiments conducted on benchmark corpora such as CNN/Daily Mail, DUC-2004, DUC-2006, and DUC-2007 demonstrate the effectiveness of the

suggested framework, as it has outperformed other state-of-the-art systems in ROUGE-1, ROUGE-2, and ROUGE-L metrics.

The most important contribution of this study is the design of a new hybrid summarization architecture called FL-BART, which incorporates fuzzy logic based extractive summary generation process with BART abstractive architecture. The proposed system improves the representation of sentences in a summary using fuzzy rules to determine the importance and novelty of sentences, which enables it to extract sentences from texts to create intermediate summaries. Also, the summaries obtained using fuzzy extractive technique are refined using the BART transformer which helps develop better summaries based on semantics, readability, and retention of information.

Outline: In Section 2, the literature review of extractive and abstractive methods used in text summarization is presented. The research objective is outlined in section 3. The process of developing the suggested hybrid framework is outlined in Section 4. Section 5 presents the dataset and experimental results. The paper concludes with some prospective directions for further research in Section 6.

2 Literature Survey

This article provides a thorough analysis of the various extractive and abstractive techniques used in text summarization.

2.1 Text Summarization Based Extractive Technique

In 1958 [15], a single document was summarized for the first time. Here, the statistical data collected from the article's word frequency was used to isolate the crucial passages. Later in 1969 [16] came to the conclusion that word frequency alone wasn't enough, and that other elements like phrases, title words, and sentence positioning should be taken into account as well. The project's use of binary classifier dates back to 1995 [17]. The selection of sentences in [18, 19] was also informed by Bayesian classifier methods. A wide variety of machine learning approaches were

Table 1. Description of Extractive text summarization

S.No	Author	Technique	Description
1	[34]	Word Frequency	The statistical data collected from the article's word frequency was used to isolate the crucial passages
2	[16]	Word Frequency + cue words	The author concluded that word frequency alone wasn't enough, and other elements like phrases, title words, and sentence positioning should be taken into account as well
3	[35]	Bayesian classifier	The author suggested Sentence selection should also be done
4	[36]	Hidden Markov	A machine learning model is applied on annotated corpora
5	[37]	Decision Tree	Decision Tree is applied on annotated dataset for text summarization
6	[38]	Deep Learning	A double layer bi-directional GRU based RNN is utilized
7	[39]	Fuzzy based	In this customized fuzzy features and neural sequence-to-sequence model with attention mechanism on the word distribution is implemented
8	[40]	Swarm Based	A novel TLBO summarization method which includes cohesion, non-redundancy and readability.
9	[41]	Graph-based summarizing framework	The authors suggested that the graph based summarizing framework can be used for both single- and multi-document summarization
10	[42]	Deep auto-encoders and sentence-level embedding's	The authors offered a unique framework (named SummCoder) for extracting text summarization of a single document
11	[27]	machine learning, and graph-based multi-document summarizer	It was a statistical, machine learning and graph based technique for extractive text summarization. It was a cluster based technique for multi-document
12	[43]	Separate and mixed topic vectors, as well as a new approach called topic vectors	The authors suggested an extractive summarizing strategy that integrates topic modelling and semantic measure

effectively applied to text summarization after the availability of annotated corpora. Hidden Markov [20], Decision Tree [19, 21, 22], and Integer linear programming [23] are some other approaches. In order to avoid repetition, [24] suggests an extractive summarizing strategy that integrates topic modelling and semantic measure. Separate and mixed topic vectors, as well as a new approach called topic vectors, were presented in this study. In [25], a paradigm for graph-extraction-based text summarization using four algorithms is proposed. The algorithm constrains the summary by picking out the most important sentences. In addition, [26] proposes a graph-based summarizing framework that may be used for both single- and multi-document summarization. In [27], the authors offer a statistical, machine learning, and graph-based extractive multi-document summarizer.

The primary focus of the DUC-2002 and DUC-2006 document was determined using a clustering-based technique.

In [28], the authors offer a unique framework (named SummCoder) for extracting text summarization of a single document.

Deep auto-encoders and sentence-level embeddings were used for this. Ranking and selection of sentences were based on three criteria: content relevancy, novelty, and place within the phrase. Also shown was the Illegal Document Summarizing (TIDSumm) dataset, a novel corpus specifically designed for the purpose of text summarization. In addition to the aforementioned strategies, extractive text summarization [29] has also been successfully implemented using other techniques like Fuzzy based [30], swarm based [31], feature relevance based [32] and deep learning based [33].

Table 2. Description of abstractive text summarization

S.No	Author	Technique	Description
1	[45]	Encoder with attention and a beam search-based decoder	A data driven model for sentence summarization. Model was trained end-to-end. Attention Mechanism is added along with a beam search decoder. DUC-2004 dataset is used
2	[60]	Encoder-decoder	The authors used a straightforward encoder-decoder paradigm for machine translation
3	[61]	Sequence-to-sequence paradigm	The authors suggested a novel architecture for abstractive summarization that builds on the sequence-to-sequence paradigm
4	[49]	Generative latent variables and discriminative deterministic states	The authors proposed an encoder-decoder based model for abstractive text summarization utilizing Generative latent variables
5	[52]	Neural Network	The authors proposed a neural network model for huge documents and summaries. They use reinforcement learning in place of traditional maximum likelihood training
6	[53]	Convolution Neural Network	The authors developed a convolutional neural network summarizing model that gives the user control over the summary's length, style, and the inclusion or exclusion of entities in which they are personally invested
7	[55]	RNN-based encoder-decoder with a pointer and attention	The authors offered a two-stage method for extracting essential information from lengthy texts (EA-LTS)
8	[50]	Generator G and a discriminator D	The authors proposed an adversarial architecture that uses a generator G and a discriminator D to increase the summary's readability
9	[54]	Dual encoder	The authors suggested a sequence-to-sequence framework extension (DEATS) for abstractive text summarization
10	[56]	Fuzzy rules + bidirectional LSTM	The authors proposed a different hybrid model for abstractive text summarization
11	[57]	Deep auto-encoders and sentence-level embedding's	The authors presented a new statistic called CSS, which combines the SARI and Rouge counts
12	[58]	Skip though + auto encoders	The authors proposed STV-BEATS which generates the extractive summary utilizing skip though for sentence embedding and autoencoders for improved semantic representation
13	[62]	Fuzzy logic+SSO	The SSO assign weight to the selected features and fuzzy rules to generate the final summary

2.2 Text Summarization Based Abstractive Technique

There are primarily three types of abstractive approaches to text summarization: (1) Structure-based (2) based on heuristics, (3) based on discourse, (4) based on deep learning [44]. Machine translation, natural language

understanding, and dialogue systems all are benefited from the implementation of deep learning models [6].

In a nutshell, [45] describes an encoder with an attention mechanism and a beam search-based decoder.

For machine translation, [46] employs a straightforward encoder-decoder paradigm, as

given by [47]. In [48], the authors suggest a novel architecture for abstractive summarization that builds on the sequence-to-sequence paradigm. The mechanism of pointing and coverage allows this to succeed where previous models failed. For abstractive text summarization, [49] makes use of generative latent variables and discriminative deterministic states and proposes an encoder-decoder based model. As an alternative, [50] proposes an adversarial architecture that uses a generator G and a discriminator D to increase the summary's readability. The document is fed into the generator G , which then optimizes its summary generation using reinforcement learning, and the summary is then compared to the ground truth using the discriminator D . Another paper [51] used an attention-based discriminator and a generator based on word embeddings to create the text. A neural network model is proposed for huge documents and summaries [52]. This model uses reinforcement learning in place of traditional maximum likelihood training. In [53], a convolutional neural network summarizing model is developed that gives the user control over the summary's length, style, and the inclusion or exclusion of entities in which they are personally invested. Using a dual encoder, [54] suggested a sequence-to-sequence framework extension (DEATS) for abstractive text summarization. In [55], the authors offer a two-stage method for extracting essential information from lengthy texts (EA-LTS). In order to narrow down potential sentences, the extraction process used a hybrid sentence similarity measure based on a combination of the sentence vector, the Levenshtein distance, and an integrated graph model. The final summary was created using an RNN-based encoder-decoder with a pointer and attention for the abstraction stage. Using the CNN/Daily mail dataset, combined DUC 2003-2004 and DUC 2006-2007 data, [56] proposes a different hybrid model for abstractive text abstractor/extractor-based summarization. Fuzzy rules are used in the extractive portion of this model, and a bidirectional LSTM is used to provide the final summary.

There was also a proposal for a hybrid paradigm of summarization/simplification (HTSS). As part of this work, [57] presented a new statistic called CSS, which combines the SARI and Rouge

counts. STV-BEATS [58] a hybrid model which generates the extractive summary utilizing skip though for sentence embedding and autoencoders for improved semantic representation and this summary is passed to the abstractive model which consists of two encoders and a decoder. Another hybrid model utilizing [59] fuzzy logic and shark smell optimization technique for improving the final generated summary.

3 Proposed Framework

In this section, hybrid FL-BART (Fuzzy Logic and BART automatic text summarization) framework is discussed with major workflows.

There are three major stages to this process. Initially, in *stage one* the dataset is cleaned using pre-processing. In the *second stage*, an extractive summary is created utilizing Fuzzy Logic and the extractive approach.

Finally, in the *third stage* the BART model is used to generate an abstractive summary based on the extracted summary. The block diagram for visualization of the workflow process and understanding of the proposed framework is shown in Figure 1.

3.1 Stage 1: Pre-Processing

Before the pre-processing stage, the input data is subjected to various operations like stemming, segmentation, stop word removal, and tokenization.

Following the extraction of the text from the documents and the removal of any data or tags that are deemed to be superfluous, a process known as segmentation is carried out in order to separate each phrase. All of the words that make up the phrases are broken down into their respective tokens (tokenization). It is termed "stop word removal," and it was used to tidy up the content by deleting the components (redundant white spaces, special characters, and so on) that do not contribute anything to the meaning of the input material.

During the training of the network, sentences that are shorter than five words are not considered. After then, a procedure known as stemming is

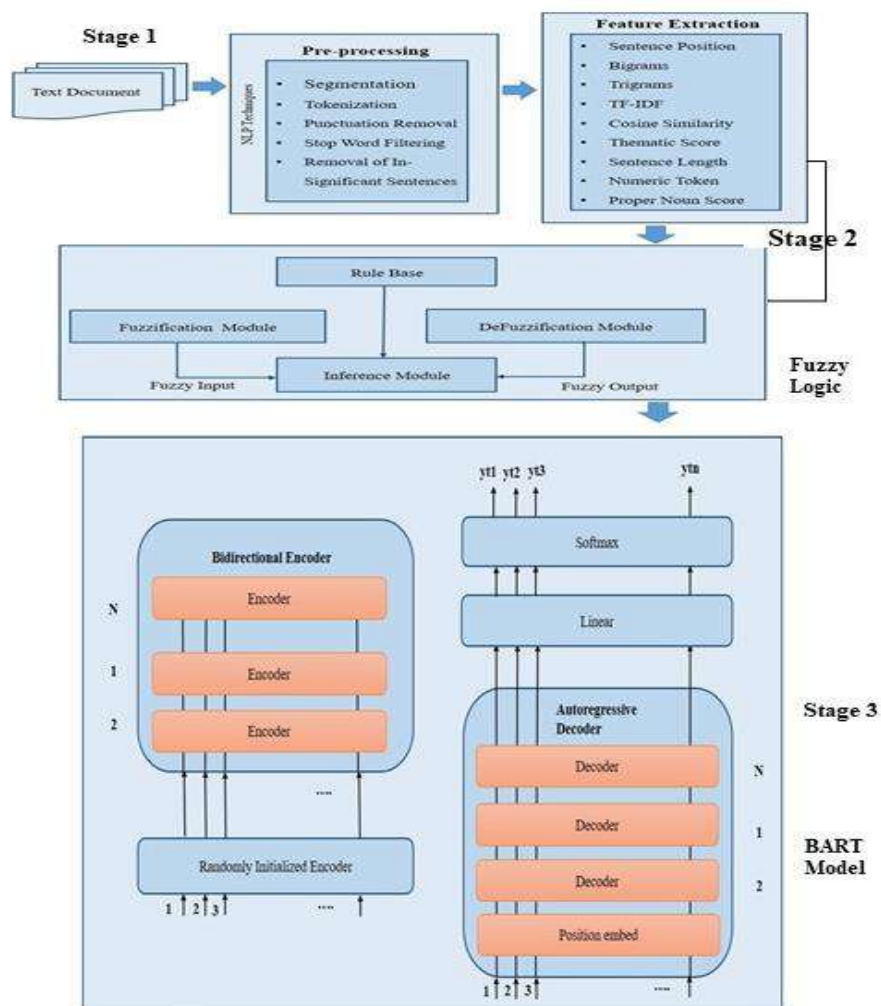


Fig. 1. The proposed hybrid FL-BART framework for text summarization

used to break each word down into its component parts.

3.2 Stage 2: Feature Extraction

A vector of features is used to represent the clean data. Such characteristics are used to rank the significance of individual sentences. The synthesis is made up of highly rated sentences. The quality of the summary can greatly benefit from careful feature selection.

Each sentence in the input data is broken down into nine features. Here are the nine characteristics [58] Sentence Position, Bigrams, Trigrams, TF-IDF, Cosine Similarity, Thematic number, Sentence Length, Proper Noun Count, Numeric Token.

3.2.1 Sentence Position

The crucial concept is that the sentence at the start or finish of the input text is more important than the other sentences in the group.

An equation is used to determine this (1):

$$\frac{\text{Sentence Position} = \text{Total Number of Sentences} - \text{Index of Sentence}}{\text{Total Number of Sentences}} \quad (1)$$

3.2.2 Bigrams

A bigram is a pair of adjacent elements drawn from a group of tokens, such as common letters, syllables, or words. For straightforward statistical analysis of text in various applications, the frequency distribution of each bigram in a string is frequently utilized.

3.2.3 Trigrams

Trigram refers to any three consecutive words in the input string. The total number of instances of groups of three consecutively different words in the text can be calculated using this method.

3.2.4 TF-IDF

Firstly, input sentence are converted into vectors followed by calculation of the TF or IDF (inverse document frequency). A statistical method to determine a word's importance in a text is the TF-IDF weight. IDF helps to reduce the weight of words that are used a lot in a set of documents but don't add anything to the meaning of the document. A given sentences can be analyzed with TF to determine the relative frequency of each word inside that sentence.

TF -relative frequency of word appearing in a document.

IDF -reversed keyword occurrence throughout documents.

TF can emphasize the importance of terms used frequently in a document. TF-IDF is calculated using equations (2-4):

$$TF(t) = \frac{\text{Number of Times term } t \text{ appear in the document}}{\text{Total Number of terms in the document}}, \quad (2)$$

$$IDF(t) = \log \frac{\text{Total Number of Documents}}{\text{Number of Documents containing the term } t}, \quad (3)$$

$$TFIDF(t) = TF(t) * IDF(t). \quad (4)$$

3.2.5 Cosine- Similarity

Prior to focusing on cosine-similarity, the centroid of the sentence vector is determined. The cosine function is then applied to each pair of words in order to determine their similarity to the centroid.

Table 3. Fuzzy Rules for extractive text summarization

IF	THEN
Sentence Position (Sps) = GOOD Proper Noun (Ns) = GOOD TF-IDF count (Fs) = GOOD	Sentence = Important
Sentence Position (Sps) = POOR Proper Noun (Ns) = POOR Numeric Tokencount(Nts)= POOR	Sentence = Unimportant
Cosine-similarity count (Csc)= GOOD Sentence Length (SlS)= GOOD	Sentence = Important
Bi-token (Bs) = GOOD Tri-token (Ts)= GOOD Proper noun (Ns)= AVERAGE Numeric token (Nts)= AVERAGE	Sentence = Average
Proper noun (Ns) = POOR Thematic count(Ths) = POOR	Sentence = Unimportant
Sentence position (Sps) = GOOD Numeric token (Nts) = AVERAGE Thematic count (Ths) = GOOD Numeric Token (Nts) = GOOD	Sentence = Important
Proper Noun (Ns) = POOR IF-IDF (Fs) = AVERAGE	Sentence = Unimportant

3.2.6 Thematic Count

Thematic words serve as passage's primary keywords. Using a technique called RAKE (Rapid Automatic Keyword Extraction), these phrases have been culled from the original document. The program prioritizes including the words it extracts in the summary.

3.2.7 Sentence Length

This is done so that the length of each sentence can be compared to the longest one in the source document. An equation is used to determine this (5):

$$\text{Sentence Length}(L) = \frac{\text{Length of Sentence}}{\text{Length of Longest Sentence}}. \quad (5)$$

3.2.8 Proper Noun Count

Proper Nouns when evaluating the value of a sentence, the presence of a proper noun is viewed as an indication of higher quality. Selecting these

sentences to use in the summary is, thus, a wise move. It's calculated by comparing the number of proper nouns in a sentence to its length and dividing by that number:

$$\text{Proper Noun Score}(N) = \frac{\text{Total number of proper nouns in sentence}}{\text{Total number of words in sentence}} \quad (6)$$

3.2.9 Numeric Token

Data given numerically in language is regarded as informative because of the clarity it provides. It is calculated by dividing the sentence's length by the total number of numerical data points:

$$\text{Numeric Token}(S) = \frac{\text{Total number of numeric data in sentence } S}{\text{Total number of words present in sentence } S} \quad (7)$$

3.3 Stage 2: Extractive Summarization Phase

The main goal of this step is to generate an intermediate extractive summary, which is achieved by first using fuzzy logic. The generated extractive summary acts as an input for the phase that comes after it.

Fuzzy inference engines draw the right conclusions for uncertainty and imprecision, which makes it suited for applications involving risks, uncertainty, and ambiguity that call for flexibility and tolerance to imprecise values. The main advantage of fuzzy logic for automated text summarization is its capacity to tolerate ambiguous linguistic information. [63] A fuzzy logic-informed method of text summarization based on ranking and sentence extraction is proposed in this research.

All sentences are ranked in terms of importance by fuzzy analyzers. They evaluate sentence relevance to find significant sentences semantically. This is an effort to condense the document into a more concise and comprehensive summary without sacrificing its essential meaning.

Due to its easy-to-understand structure of min-max operations, the Mamdani fuzzy inference system (FIS) is the most popular choice for practical use [56]. Membership functions generated by Mamdani type inference are expected to be fuzzy sets. Before this FIS can

Algorithm 1: Proposed Hybrid Model

Input : Text

Output: Summarized Text

Step 1: The Input Document are Pre-processed (Tokenization, Segmentation, Stop word and Punctuation

Removal etc).

Step 2: Word2Vec Pretrained word embedding is used to prepare a table

Step 3: Identify the features and calculate the feature count.

Nts = Numeric Token Count

Sps = Calculate Sentence Position

Bs = Bigrams

Ts = Trigrams

Ns = Proper Noun Count

Fs = TF-IDF

Csc = Cosine Similarity

Ths = Thematic Count

Sls = Sentence Length

Step 4 : Consequent Object hold universe

variable(Sps, Bs, Ts, Fs, Csc, Ths, Sls, Ns, Nts)

Step 5: Fuzzy Logic System based on Fuzzy variable & rules is utilized to calculate sentence count.

Step 6: Extractive Output based on Fuzzy system is generated.

Step 7: Previous Output is fed as input to BART Model.

Step 8: Final output summary is generated.

Step 9: Output summary is compared against reference summary and Rouge count is calculated.

compute the result given the inputs, six stages must be finished:

Selecting a set number of high-scoring sentences generates a summarization. Fuzzy sentence analyzers consider multiple factors for determining a sentence's relative importance within a text. Due to the lack of a unique formula describing the interplay of the aforementioned elements, several if-then rules (termed "fuzzy rules") have been derived. Table 3 shows the Fuzzy Rules for extractive text summarization.

3.4 Stage 3: Abstractive Summarization Phase

The BART (Bidirectional and Auto-Regressive Transformers) is a neural network architecture. BART has performed well in natural language processing (NLP), including summarizing text,

Table 4. Characteristics of datasets

Dataset	Domain	Document count	Summary Limit
DUC 2004	News	500	100 words
DUC2006	News	1300	250 words
DUC 2007	News	1200	250 words
CNN/DailyMail Dataset	News	2,87,000	-

making up new language, answering questions. BART utilizes a bidirectional encoder and a decoder that employs autoregressive techniques.

Since BART has a bidirectional encoder, the first component of BART can use it to determine the best possible representation of the input sequence.

Each token in the input text sequence receives an embedding vector from the BART encoder, and each input text sequence also receives an additional vector with sentence-level information. The BART models and training goals are partitioned into four steps:

3.4.1 Architecture

The BART architecture is made up of an encoder-decoder system, which is similar to the sequence-to-sequence models used in machine translation:

- The encoder is self-regressive, which means it creates new tokens based on the ones it already created. The encoder can take in data from both left and right because it works in both directions.
- Both the encoder and the decoder are made up of many layers of feed-forward neural networks and self-attention processes.

3.4.2 Pre-training Objectives

- Denoising Autoencoder: A denoising auto encoder goal is used to train BART before it starts using it. In this setup, random hiding of some tokens messes up the input sequences, and the model is taught to put back together the original sequences.
- Sequence-to-Sequence Learning: BART also uses a sequence-to-sequence learning goal, in which the model learns to turn source

sequences into target sequences. This helps the model understand the structure and meaning of pairs of input and output.

3.4.3 Tokenization

Like other Transformer-based models, BART breaks up raw text into sub word units using Byte-Pair Encoding (BPE) or Sentence Piece. Some tokens in the input sequences are randomly hidden during pre-training, and the model is taught to guess what those hidden tokens are.

3.4.4 Fine-Tuning

BART can perform better after pre-training. After fine-tuning a model, the changes can be performed on its settings on task-specific datasets for summarizing, translating, creating text, and more.

Depending on the type of task, fine-tuning usually includes adding more task-specific goals or loss functions.

In general, the BART model is a strong and adaptable framework for many different NLP jobs.

It effectively captures complex linguistic patterns and produces high-quality text thanks due to its use of auto-regressive decoding, bidirectional encoding, and good pre-training goals.

4 Experimental Setup

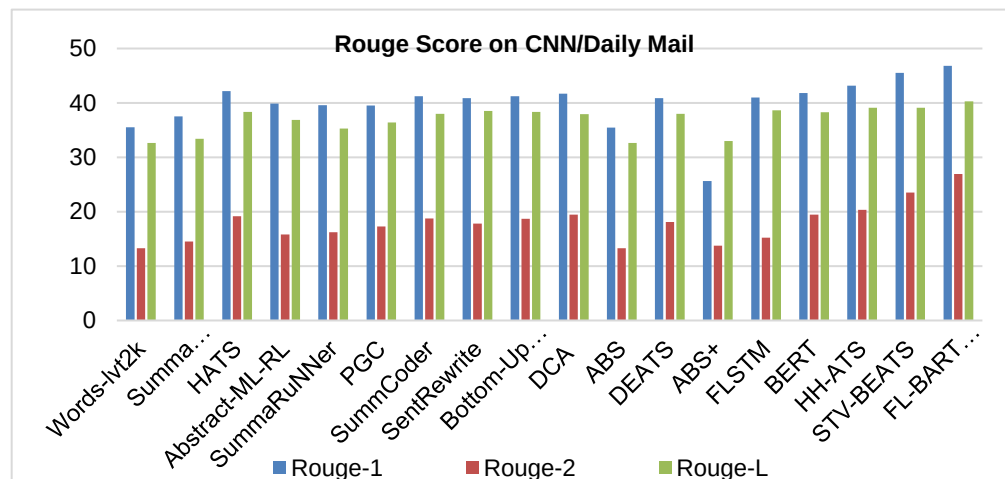
For the purpose of analyzing the effectiveness of the proposed multi-document summarizing method, this section depicts the dataset description, experimental setting and result analysis. All of the tests and changes are performed on a Windows system with an Intel 2.50 GHz CPU and 4GB of memory.

4.1 Dataset

Summaries can be generated from a number of different standard datasets. CNN/Daily Mail dataset [13] is used for conducting the experiments, which acts as a basic foundation for generic summarization of text. A multi-sentence summary of the news article is included for reference. Combining the CNN and Daily Mail datasets increases the number of available data examples for training.

Table 5. Performance evaluation of state-of-art models on CNN/DM dataset

Model	Rouge-1	Rouge-2	Rouge-L
<i>Words-lvt2k-temp-att</i>	35.5	13.30	32.65
<i>SummaRuNNer-abs</i>	37.5	14.50	33.40
<i>HATS</i>	42.16	19.17	38.35
<i>Abstract-ML-RL</i>	39.87	15.82	36.9
<i>SummaRuNNer</i>	39.6	16.2	35.3
<i>PGC</i>	39.53	17.28	36.38
<i>SummCoder</i>	41.20	18.78	37.96
<i>SentRewrite</i>	40.88	17.8	38.54
<i>Bottom-Up Summarization</i>	41.22	18.68	38.34
<i>DCA</i>	41.69	19.47	37.92
<i>ABS</i>	35.46	13.30	32.65
<i>DEATS</i>	40.85	18.08	37.96
<i>ABS+</i>	25.63	13.75	33.01
<i>FLSTM</i>	40.96	15.22	38.63
<i>BERT</i>	41.82	19.48	38.3
<i>HH-ATS</i>	43.16	20.32	39.10
<i>STV-BEATS</i>	45.53	23.52	39.10
<i>FL-BART (Proposed)</i>	46.81	26.92	40.27

**Fig. 2.** FL-BART performance on the CNN/Daily Mail dataset is compared to that of state-of-art models

Separate sets are used for training, testing, and analyzing the merged dataset. There was a total of 311,672 articles utilized across the training, testing, and validation phases, with a total of 249,337 used for training.

The National Institute of Standards and Technology (NIST) produced DUC-2006 and DUC-2007 are also utilized for implementation. Both the corpus and the testing data are used to assess the quality of the summary.

Table 4 provides details on the text summarizing datasets available for research. ROUGE (Recall-Oriented Understudy for Gisting Evaluation) [64] is used to evaluate the summaries by counting how many words are shared between the gold standard summary and the summaries obtained by the evaluation system.

4.2 Experimental Setting

When training and fine-tuning the model, the CNN/Daily Mail dataset is employed.

The size of the embeddings is fixed at 2400 because at that resolution; the results obtained by a number of researchers represented the state of the art. To generate embeddings of varying sizes, retraining the model is required.

The created embeddings then need to be tested on a variety of natural language processing tasks before being used. The word embedding dimension is set to 128, whereas the hidden state dimension is 256. The system defaults to a vocabulary size of 50k words. Setting the learning rate to 0.1.

When used for both development and testing, the token size is held constant at 120. This is done to facilitate learning, and the output summary is decoded using beam search with a beam size of 4.

The parameters of the network are initially seeded at random from a uniform distribution between [0.05, 0.05].

Additionally, the Adagrad algorithm is utilized for enhancement. The model's flexibility for use with a smaller vocab size is due, in part, to its handling of OOV terms on its own.

5 Result Analysis and Discussion

This section analyzes and compares the proposed framework FL-BART against other state-of-art models on the specified datasets.

5.1 CNN/Daily Mail Dataset

Abstractive text summarization using Attentional Encoder Decoder [45] was proposed to solve the critical problems such as modeling key-words, and other models like SummaRunner-abs [33] is an

RNN based model that has been transformed from an extractive model.

It captures the hierarchy of sentence-toward structure for improvement in performance. Other abstractive techniques [65] focus on critical problems basic architecture, such as modeling key-words, capturing the hierarchy of sentence-toward structure, improvement in performance. ABS and ABS+ [45] on the CNN/Daily Mail dataset are a model built using a standard encoder-decoder with an attention mechanism. Sequence-to-sequence models with attention, such as seq2seq+atten [47], offer another option.

While PGC [48] is a tried-and-true sequence-to-sequence model with coverage to get around the OOV word problem. The Hybrid Approach to Simulation (HATS) [66] is a learning paradigm that attempts to mimic the human learning process.

Additionally, FLSTM [56], an ensemble model that makes use of the fuzzy and LSTM approach, is compared to it. Human-like deep network HH-AT [67] is made up of a knowledge module, a multi task module.

Two models for abstractive text summarizing STV-BEATS [58], and DEATS [54], which consists of two encoders and a decoder—have been developed.

5.2 DUC-2004

A topic detection method called TOPIARY [68], ABS and ABS+ [45], the first machine translation model; RAS-Elman [46], an encoding and decoding system where the encoder makes use of attention and the decoder relies on a recurrent neural network.

Another model that relies on attention is wordslvt5k-lsent [65], while SEASS [69] employs a selective gate network in its encoder-decoder.

Methods like SummCoder [28] for extractive summarization of texts. DEATS [54] is a model which utilizes two encoders, (primary, secondary) and one decoder for abstractive text summarization.

A hybrid framework (STV-BEATS) which is a conjunction of extractive model and abstractive model. For extractive phase skip thought vector for sentence embedding and LSTM based autoencoder is used. The abstractive phase consists of bi-encoder and a decoder [58].

Table 6. Performance evaluation of state-of-art models on DUC-2004 dataset

Method	Rouge-1	Rouge-2	Rouge-L
<i>ABS</i>	26.55	7.06	22.05
<i>LSA</i>	29.73	3.99	26.34
<i>ABS+</i>	28.18	8.49	23.81
<i>RAS-ELMAN</i>	28.97	8.26	24.06
<i>SummCoder</i>	29.31	9.54	25.24
<i>Words-lvt5k-lsent</i>	28.61	9.42	25.24
<i>LexRank</i>	31.21	5.25	25.01
<i>SEASS</i>	29.21	9.56	25.51
<i>RBM</i>	25.46	8.27	42.38
<i>DEATS</i>	29.91	9.61	25.95
<i>TOPIARY</i>	25.12	6.46	20.12
<i>STV-BEATS</i>	35.91	8.96	30.38
<i>TextRank</i>	36.21	7.39	27.91
<i>FL-BART (Proposed)</i>	43.11	25.21	31.7

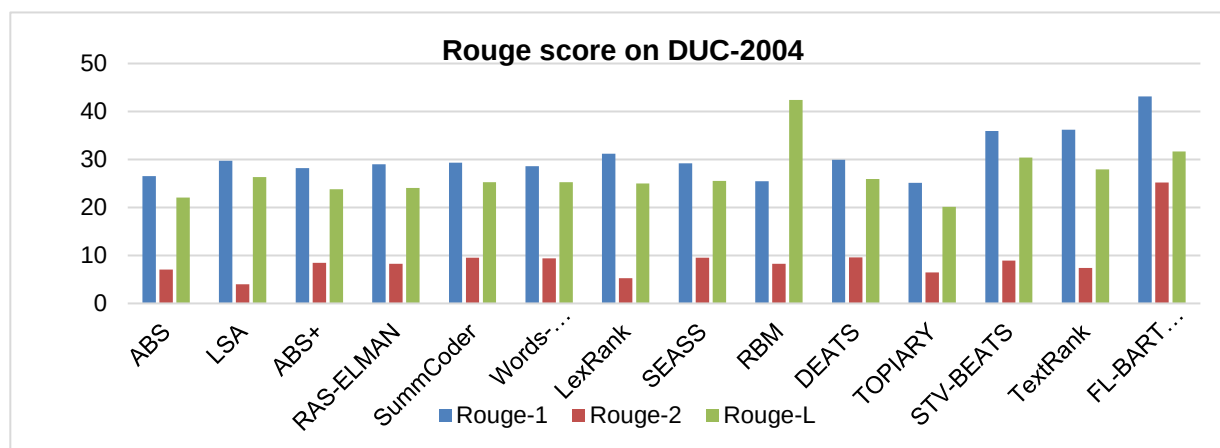
**Fig. 3.** FL-BART performance on the DUC-2004 dataset is compared to that of cutting-edge models

Table 5 and Fig. 2 represent experimental data from FL-BART experiments run on the CNN/Daily Mail dataset. As compared to state-of-art models, it performs exceptionally well on CNN/Daily Mail Dataset. The Rouge score of proposed (FL-BART) model are **Rouge-1** count is 46.81, **Rouge-2** count is 26.92, and **Rouge-L** count is 40.27.

The experimental results show that the proposed model outperforms a number of state-of-

art models. Table 4 and Fig. 3 display the experimental results of the evaluation of DUC-2004, which shows high Rouge scores. The **Rouge-1** count of 43.11, **Rouge-2** count of 25.21 and **Rouge-L** count of 31.7.

However, the proposed model is also experimented with DUC-2006 and DUC-2007 datasets. These are majorly utilized for extractive text summarization. It also performs well on the

Table 7. Result of Proposed Framework (FL-BART) on various dataset

	Rouge-1	Rouge-2	Rouge-L
DUC2004	46.31	26.53	40.27
DUC2006	43.12	25.34	31.77
DUC2007	38.92	14.96	25.52
CNN/DailyMail	46.82	26.92	40.27

DUC-2006 and DUC-2007 datasets, with a good Rouge count as shown in Table 7. The **Rouge-1** count is 43.12, the **Rouge-2** count is 25.34, and the **Rouge-L** count is 31.77 on DUC 2006 Dataset and the **Rouge-1** count is 38.92, **Rouge-2** count is 14.96, and the **Rouge-L** count is 25.52 on DUC 2007 Dataset.

The experiment shows in Table 7 that the CNN/Daily Mail dataset gives a better overall Rouge count as the size of the CNN/DailyMail dataset is larger in size than DUC dataset. The abstractive model always performs better with larger dataset. This is why the CNN/Daily Mail dataset is a good choice for training the model. Qualitative testing across multiple data sets confirms its usefulness.

6 Conclusion and Future Work

In this paper, a model FL-BART is proposed, a hybrid extractive abstractive based framework, for automatic text summarization; it improves upon the standard extractive and abstractive approach. As a means of improving the semantic representation of sentences, it creates extractive summaries using Fuzzy Rules. The BART model uses the extractive summary as input to produce the abstractive summary. The efficiency of both the extractive and abstractive models is increased by their union. On the Rouge metric, experimental results on the CNN/Daily Mail dataset demonstrate that the proposed model performs better than state-of-art models. ROUGE-1 and ROUGE-2 scores also performs better while testing on DUC-2004 datasets.

The ROUGE-L score produced is comparative to that of other models. The proposed hybrid model

is also experimented on DUC-2006 and DUC-2007 datasets which is majorly used for extractive summarization. In future, F1 count can be raised by utilizing a different extractive model trained with sentence embedding.

This can be combined with additional abstractive models to boost the hybrid's performance, such as reinforcement, transformers, and hybrid training functions. Further, the use of transformer-based approaches in creating better embedding models like BERT, SBERT, or even domain-specific embeddings could help in increasing the accuracy and F1-score. In addition, advancements could be made through reinforcement learning techniques along with the adaptive hybrid approach to train the extractive and abstractive components. The proposed work can be investigated deeply on developing methods for multi-lingual and domain-specific summarization.

Also, utilizing explainable AI approaches and implementing light-weighted systems could improve the efficiency of the proposed system.

References

1. Matias-Mendoza, G.A., Ledeneva, Y., Garcia-Hernandez, R.A. (2024). Spanish Automatic Text Summarization: A Survey. *Computacion y Sistemas*, Vol. 28, No. 3, pp. 1361–1376.
2. Akhmetov, I., Nurlybayeva, S., Ualiyeva, I. (2023). A Comprehensive Review on Automatic Text Summarization. *Computacion y Sistemas*, Vol. 27, No. 4, pp. 1203–1240. DOI: 10.13053/cys-27-4-4792.
3. Lloret, E., Palomar, M. (2012). Text Summarisation in Progress: A Literature Review. *Artificial Intelligence Review*, Vol. 37, No. 1, pp. 1–41. DOI: 10.1007/s10462-011-9244-y.
4. Verma, R.M., Lee, D. (2017). Extractive Summarization: Limits, Compression, Generalized Model and Heuristics. *Computación y Sistemas*, Vol. 21. DOI: 10.13053/cys-21-4-2885.
5. Gambhir, M., Gupta, V. (2017). Recent Automatic Text Summarization Techniques: A

- Survey. *Artificial Intelligence Review*, Vol. 47, No. 1, pp. 1–66. DOI: 10.1007/s10462-016-9500-z.
6. **Young, T., Hazarika, D., Poria, S. (2018).** Recent Trends in Deep Learning Based Natural Language Processing. *IEEE Computational Intelligence Magazine*, Vol. 13, No. 3, pp. 55–75. DOI: 10.1109/mci.2018.2840738.
 7. **Schmidt, R.M. (2019).** Recurrent Neural Networks (RNNs): A Gentle Introduction and Overview. *arXiv Preprint arXiv:1912.05911*. DOI: 10.48550/arxiv.1912.05911.
 8. **Kumar, A., Sharma, A. (2019).** Systematic Literature Review of Fuzzy Logic Based Text Summarization. *Iranian Journal of Fuzzy Systems*, Vol. 16, No. 5, pp. 45–59.
 9. **Chipman, H.A., George, E.I., McCulloch, R.E. (2010).** BART: Bayesian Additive Regression Trees.
 10. **Tan, Y.V., Roy, J. (2019).** Bayesian Additive Regression Trees and the General BART Model. *Statistics in Medicine*, Vol. 38, No. 25, pp. 5048–5069. DOI: 10.1002/sim.8347.
 11. **Blumenthal, M., Luo, G., Schilling, M. (2023).** Deep, Deep Learning With BART. *Magnetic Resonance in Medicine*, Vol. 89, No. 2, pp. 678–693. DOI: 10.1002/mrm.29463.
 12. **McCulloch, R., Sparapani, R., Gramacy, R. (2018).** BART: Bayesian Additive Regression Trees. *R Package Version*, Vol. 1.
 13. **Hermann, K.M., Kocisky, T., Grefenstette, E. (2015).** Teaching Machines to Read and Comprehend. *Advances in Neural Information Processing Systems*, Vol. 28, pp. 1693–1701.
 14. **Lin, C.-Y., Hovy, E. (2003).** Automatic Evaluation of Summaries Using N-Gram Co-Occurrence Statistics. *Proceedings of the 2003 Human Language Technology Conference of the North American Chapter of the Association for Computational Linguistics*, pp. 150–157.
 15. **Luhn, H.P. (1958).** The Automatic Creation of Literature Abstracts. *IBM Journal of Research and Development*, Vol. 2, No. 2, pp. 159–165. DOI: 10.1147/rd.22.0159.
 16. **Edmundson, H.P. (1969).** New Methods in Automatic Extracting. *Journal of the ACM* (JACM), Vol. 16, No. 2, pp. 264–285. DOI: 10.1145/321510.321519.
 17. **Kupiec, J., Pedersen, J., Chen, F. (1995).** A Trainable Document Summarizer. *Proceedings of the 18th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, pp. 68–73. DOI: 10.1145/215206.215333.
 18. **Teufel, S. (1997).** Sentence Extraction as a Classification Task. *Intelligent Scalable Text Summarization*.
 19. **Neto, J.L., Freitas, A.A., Kaestner, C.A.A. (2002).** Automatic Text Summarization Using a Machine Learning Approach. *Brazilian Symposium on Artificial Intelligence*, pp. 205–215. DOI: 10.1007/3-540-36127-8_20.
 20. **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*, pp. 406–407. DOI: 10.1145/383952.384042.
 21. **Mani, I., Bloedorn, E. (1998).** Machine Learning of Generic and User-Focused Summarization. *AAAI/IAAI*, pp. 821–826.
 22. **Knight, K., Marcu, D. (2002).** Summarization beyond Sentence Extraction: A Probabilistic Approach to Sentence Compression. *Artificial Intelligence*, Vol. 139, No. 1, pp. 91–107. DOI: 10.1016/s0004-3702(02)00222-9.
 23. **Luo, W., Liu, F., Liu, Z. (2018).** A Novel ILP Framework for Summarizing Content with High Lexical Variety. *arXiv Preprint arXiv:1807.09671*. DOI: 10.48550/arxiv.1807.09671.
 24. **Belwal, R.C., Rai, S., Gupta, A. (2021).** Text Summarization Using Topic-Based Vector Space Model and Semantic Measure. *Information Processing & Management*, Vol. 58, No. 3. DOI: 10.1016/j.ipm.2021.102536.
 25. **El-Kassas, W.S., Salama, C.R., Rafea, A.A. (2020).** EdgeSumm: Graph-Based Framework for Automatic Text Summarization. *Information Processing & Management*, Vol. 57, No. 6. DOI: 10.1016/j.ipm.2020.102264.
 26. **Mohamed, M., Oussalah, M. (2019).** SRL-ESA-TextSum: A Text Summarization Approach Based on Semantic Role Labeling

- and Explicit Semantic Analysis. *Information Processing & Management*, Vol. 56, No. 4, pp. 1356–1372. DOI: 10.1016/j.ipm.2019.04.003.
27. **Bidoki, M., Moosavi, M.R., Fakhrahmad, M. (2020).** A Semantic Approach to Extractive Multi-Document Summarization: Applying Sentence Expansion for Tuning of Conceptual Densities. *Information Processing & Management*, Vol. 57, No. 6. DOI: 10.1016/j.ipm.2020.102341.
 28. **Joshi, A., Fidalgo, E., Alegre, E. (2019).** SummCoder: An Unsupervised Framework for Extractive Text Summarization Based on Deep Auto-Encoders. *Expert Systems with Applications*, Vol. 129, pp. 200–215. DOI: 10.1016/j.eswa.2019.04.014.
 29. **El-Kassas, W.S., Salama, C.R., Rafea, A.A. (2021).** Automatic Text Summarization: A Comprehensive Survey. *Expert Systems with Applications*. DOI: 10.1016/j.eswa.2020.113679.
 30. **Sahba, R., Ebadi, N., Jamshidi, M. (2018).** Automatic Text Summarization Using Customizable Fuzzy Features and Attention on the Context and Vocabulary. 2018 World Automation Congress (WAC), pp. 1–5. DOI: 10.1109/wac.2018.8430373.
 31. **Verma, P., Om, H. (2019).** MCRMR: Maximum Coverage and Relevancy with Minimal Redundancy Based Multi-Document Summarization. *Expert Systems with Applications*, Vol. 120, pp. 43–56. DOI: 10.1016/j.eswa.2018.11.015.
 32. **Neri-Mendoza, V., others (2024).** Multi-Document Text Summarization through Features Relevance Calculation. *Computacion y Sistemas*, Vol. 28, No. 3, pp. 1417–1427. DOI: 10.13053/cys-28-3-5201.
 33. **Nallapati, R., Zhai, F., Zhou, B. (2017).** Summarunner: A Recurrent Neural Network Based Sequence Model for Extractive Summarization of Documents. Thirty-First AAAI Conference on Artificial Intelligence.
 34. **Luhn, H.P. (1958).** The Automatic Creation of Literature Abstracts. *IBM Journal of Research and Development*, Vol. 2, No. 2, pp. 159–165. DOI: 10.1147/rd.22.0159.
 35. **Teufel, S. (1997).** Sentence Extraction as a Classification Task. *Intelligent Scalable Text Summarization*.
 36. **Conroy, J.M., O’Leary, D.P. (2001).** Text Summarization via Hidden Markov Models. *SIGIR*, pp. 406–407. DOI: 10.1145/383952.384042.
 37. **Knight, K., Marcu, D. (2002).** Summarization beyond Sentence Extraction. *Artificial Intelligence*, Vol. 139, No. 1, pp. 91–107. DOI: 10.1016/s0004-3702(02)00222-9.
 38. **Nallapati, R. (2016).** Abstractive Text Summarization Using Sequence-to-Sequence RNNs and Beyond. *arXiv Preprint arXiv:1602.06023*. DOI: 10.48550/arxiv.1602.06023.
 39. **Sahba, R. (2018).** Automatic Text Summarization Using Customizable Fuzzy Features. *WAC*, pp. 1–5. DOI: 10.1109/wac.2018.8430373.
 40. **Verma, P., Om, H. (2019).** A Novel Approach for Text Summarization Using Optimal Combination of Sentence Scoring Methods. *Sadhana*, Vol. 44, No. 5. DOI: 10.1007/s12046-019-1100-3.
 41. **Mohamed, M., Oussalah, M. (2019).** SRL-ESA-TextSum. *Information Processing & Management*, Vol. 56, No. 4, pp. 1356–1372. DOI: 10.1016/j.ipm.2019.04.003.
 42. **Joshi, A., Fidalgo, E., Alegre, E. (2019).** SummCoder: An Unsupervised Framework for Extractive Text Summarization Based on Deep Auto-Encoders. *Expert Systems with Applications*, Vol. 129, pp. 200–215. DOI: 10.1016/j.eswa.2019.04.014.
 43. **Belwal, R.C., Rai, S., Gupta, A. (2021).** Text Summarization Using Topic-Based Vector Space Model and Semantic Measure. *Information Processing & Management*, Vol. 58, No. 3. DOI: 10.1016/j.ipm.2021.102536.
 44. **Gupta, S., Gupta, S.K. (2019).** Abstractive Summarization: An Overview of the State of the Art. *Expert Systems with Applications*, Vol. 121, pp. 49–65. DOI: 10.1016/j.eswa.2018.12.011.
 45. **Rush, A.M., Chopra, S., Weston, J. (2015).** A Neural Attention Model for Abstractive Sentence Summarization. *arXiv Preprint*

- arXiv:1509.00685. DOI: 10.48550/arxiv.1509.00685.
46. **Chopra, S., Auli, M., Rush, A.M. (2016).** Abstractive Sentence Summarization with Attentive Recurrent Neural Networks. Proceedings of the 2016 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, pp. 93–98. DOI: 10.18653/v1/n16-1012.
 47. **Bahdanau, D., Cho, K., Bengio, Y. (2014).** Neural Machine Translation by Jointly Learning to Align and Translate. arXiv Preprint arXiv:1409.0473. DOI: 10.48550/arxiv.1409.0473.
 48. **See, A., Liu, P.J., Manning, C.D. (2017).** Get to the Point: Summarization with Pointer-Generator Networks. arXiv Preprint arXiv:1704.04368. DOI: 10.48550/arxiv.1704.04368.
 49. **Li, P., Lam, W., Bing, L. (2017).** Deep Recurrent Generative Decoder for Abstractive Text Summarization. arXiv Preprint arXiv:1708.00625. DOI: 10.48550/arxiv.1708.00625.
 50. **Liu, L., Lu, Y., Yang, M. (2018).** Generative Adversarial Network for Abstractive Text Summarization. Proceedings of the AAAI Conference on Artificial Intelligence.
 51. **Rizzo, G., Van, T.H.M. (2020).** Adversarial Text Generation with Context Adapted Global Knowledge and a Self-Attentive Discriminator. Information Processing & Management, Vol. 57, No. 6. DOI: 10.1016/j.ipm.2020.102217.
 52. **Paulus, R., Xiong, C., Socher, R. (2017).** A Deep Reinforced Model for Abstractive Summarization. arXiv Preprint arXiv:1705.04304. DOI: 10.48550/arxiv.1705.04304.
 53. **Fan, A., Grangier, D., Auli, M. (2017).** Controllable Abstractive Summarization. arXiv Preprint arXiv:1711.05217. DOI: 10.48550/arxiv.1711.05217.
 54. **Yao, K., Zhang, L., Du, D. (2018).** Dual Encoding for Abstractive Text Summarization. IEEE Transactions on Cybernetics, Vol. 50, No. 3, pp. 985–996. DOI: 10.1109/tcyb.2018.2882583.
 55. **Wang, S., Zhao, X., Li, B. (2017).** Integrating Extractive and Abstractive Models for Long Text Summarization. 2017 IEEE International Congress on Big Data (BigData Congress), pp. 305–312. DOI: 10.1109/bigdatacongress.2017.48.
 56. **Tomer, M., Kumar, M. (2020).** Improving Text Summarization Using Ensembled Approach Based on Fuzzy With LSTM. Arabian Journal for Science and Engineering, Vol. 45, No. 12, pp. 10743–10754. DOI: 10.1007/s13369-020-04803-x.
 57. **Zaman, F., Shardlow, M., Hassan, S.-U. (2020).** HTSS: A Novel Hybrid Text Summarisation and Simplification Architecture. Information Processing & Management, Vol. 57, No. 6. DOI: 10.1016/j.ipm.2020.102351.
 58. **Tomer, M., Kumar, M. (2022).** STV-BEATS: Skip Thought Vector and Bi-Encoder Based Automatic Text Summarizer. Knowledge-Based Systems, Vol. 240. DOI: 10.1016/j.knosys.2021.108108.
 59. **Tomer, U.T.M., Kumar, M., Hashmi, A. (2023).** Enhancing metaheuristic based extractive text summarization with fuzzy logic. Neural Computing and Applications, Vol. 35, No. 13, pp. 9711–9723. DOI: 10.1007/s00521-023-08209-5.
 60. **Chopra, S., Auli, M., Rush, A.M. (2016).** Abstractive Sentence Summarization with Attentive Recurrent Neural Networks. NAACL-HLT, pp. 93–98. DOI: 10.18653/v1/n16-1012.
 61. **See, A., Liu, P.J., Manning, C.D. (2017).** Get to the Point: Summarization with Pointer-Generator Networks. arXiv Preprint arXiv:1704.04368. DOI: 10.48550/arxiv.1704.04368.
 62. **Tomer, M., Kumar, M., Hashmi, A. (2023).** Title Not Available. Neural Computing and Applications, Vol. 35, No. 13, pp. 9711–9723. DOI: 10.1007/s00521-023-08209-5.
 63. **Goularte, F.B., Nassar, S.M., Fileto, R. (2019).** A Text Summarization Method Based on Fuzzy Rules and Applicable to Automated Assessment. Expert Systems with Applications, Vol. 115, pp. 264–275. DOI: 10.1016/j.eswa.2018.08.006.

64. **Lin, C.-Y. (2004).** ROUGE: A Package for Automatic Evaluation of Summaries. Workshop on Text Summarization Branches Out (WAS 2004).
65. **Nallapati, R., Zhou, B., Gulcehre, C. (2016).** Abstractive Text Summarization Using Sequence-to-Sequence RNNs and Beyond. arXiv Preprint arXiv:1602.06023. DOI: 10.48550/arxiv.1602.06023.
66. **Yang, M., Qu, Q., Tu, W. (2019).** Exploring Human-Like Reading Strategy for Abstractive Text Summarization. Proceedings of the AAAI Conference on Artificial Intelligence, pp. 7362–7369. DOI: 10.1609/aaai.v33i01.33017362.
67. **Liu, Y., Lapata, M. (2019).** Hierarchical Transformers for Multi-Document Summarization. arXiv Preprint arXiv:1905.13164. DOI: 10.48550/arxiv.1905.13164.
68. **Zajic, D., Dorr, B., Schwartz, R. (2004).** Bbn/umd at Duc-2004: Topiary. Proceedings of the HLT-NAACL 2004 Document Understanding Workshop, pp. 112–119.
69. **Zhou, Q., Yang, N., Wei, F. (2017).** Selective Encoding for Abstractive Sentence Summarization. arXiv Preprint arXiv:1704.07073. DOI: 10.48550/arxiv.1704.07073.

Article received on 05/02/2026; accepted on 21/04/2026.

**Corresponding author is Charu Gupta.*