Deep Learning for Automatic Text Summarization: Introductory part of Current Techniques

[1] Divya Amade and [2] Rajshri Lanjewar [3] Vivek Kumar Sinha

[1] M. Tech Student, Dept. of CSE, Raipur Institute of Technology, Raipur, Chhattisgarh, India

[2] Assistant Professor, Dept. of CSE, Raipur Institute of Technology, Raipur, Chhattisgarh, India

[3] Head Of Dept. of CSE, Raipur Institute of Technology, Raipur, Chhattisgarh, India

Abstract:

A crucial job in natural language processing (NLP) is text summarization,s that aims to condense large volumes of text into shorter summaries while retaining the key information. Deep learning techniques have recently shown promising results in text summarization, outperforming traditional machine learning approaches. In this study, we give a thorough assessment of current studies on deep learning-based text summarization. We explore the most widely used deep learning architectures for text summarization, including transformer-based models, recurrent neural networks, and convolutional neural networks. We also review various attention mechanisms used to improve the quality of generated summaries. Additionally, we present an overview of recent datasets and evaluation metrics used to benchmark text summarization systems. We also discuss current challenges and future directions for research in text summarization using deep learning techniques, including the need for more diverse datasets and better evaluation metrics.

Keywords: **Deep learning, LSTM, sequence-to-sequence models, RNN, GRU, reinforcement learning, transformers**.

1. Introduction:

Text summarising is a crucial NLP job that involves generating a concise summary of a large piece of text while retaining its most important information. It is widely used in various applications, such as news articles, academic papers, and legal documents. Traditional text summarization techniques rely on heuristic-based algorithms and statistical models, which may not always capture the nuances of human language.

In recent years, deep learning techniques have shown great promise in text summarization due to their ability to learn complex representations of text data. Deep learning models can effectively capture the contextual and semantic relationships between words and phrases, making it possible to generate high-quality summaries that are both accurate and concise.

[1] H.P. Luhn developed text summarization in the 1950s, and the IBM 701 was the first commercial computer to use it, He calculated the frequencies of the most common phrases based on their occurrence using the bag-of-words method. The most common terms were then chosen, and each phrase was given a number based on a regular occurrence. Natural language processing (NLP) was gradually used to examine linguistics and to start employing different word kinds. Text extraction, classification, and categorization are the main objectives. Following that, the deconstruction of sentences into vectors and words into their fundamental forms was explored with the growth of NLP between 1990 and 2000. The ATS field has made great strides thanks to contemporary deep learning methods including recurrent neural networks (RNN) [5] and long short-term memory (LSTM) [6], as well as cutting-edge

International Journal of Scientific Research in Engineering and Management (IJSREM)
Volume: 07 Issue: 07 | July - 2023 SJIF Rating: 8.176 ISSN: 2582-3930

NLP methods like neural word embedding [2], bag of words (BoW) [3], and word2vec [4. The history of ATS development from the 1950s to the present is examined in this essay.

Text summarization procedures used from the 1970s to the early 2000s are recognised as traditional methods. A better comprehension of the content is necessary to identify the critical keywords for conventional text summing algorithms. Due to its significant contribution to the research and growth of automation, ATS has grown in popularity [7][8][9].

The Three Main Text Summarization Techniques Are Illustrated in Figure 1 Below.

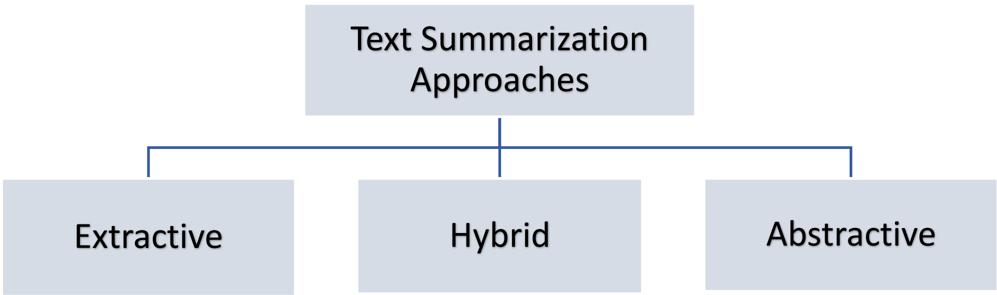

Fig1:- Automatic Text Summarization Approaches

- **Extractive:-** The extractive technique creates the summary by picking and choosing words or sentences from the source material. The system evaluates variables including word frequency, sentence location, and relevance to the entire manuscript to determine the value of each phrase. The summary is then created by extracting and joining the top-ranked sentences.

- **Hybrid:-** The hybrid approach combines elements of both extractive and abstractive methods to leverage their respective strengths. It involves applying extractive techniques to identify important sentences or phrases as a foundation, and then using abstractive techniques to paraphrase and refine the extracted content, creating a more coherent and concise summary.

- **Abstractive:-** In the abstractive approach, the system attempts to understand the meaning of the source text and generate a summary using natural language generation techniques. It involves analyzing the input text, identifying important concepts, and then generating concise and coherent sentences that capture the essence of the original content. This method relies on advanced NLP models, such as neural networks and deep learning architectures.

2 Literature Review:-

[10] "Arabic text summarization using deep learning approach" was the idea put forth by Molham Al Maleh et al. They created an Arabic dataset of headlines from summarised articles. There are about 300,000 entries in this data collection. After that, they compare the outcomes utilising the ROUGE scale and baseline summarization models to the earlier data set.

[11] According to Shengli Song et al., By taking a closer look at these more finely tuned, smaller, semantic phrase-like units than individual words, LSTM-CNN based ATS framework (ATSDL) may be able to generate new sentences. A text summarization technique called ATSDL consists of two fundamental steps: the first extracts terms from source sentences, and the second employs deep learning to generate text summaries. This sets it apart from other abstraction-based strategies now in use. Experimental results on the CNN and DailyMail datasets show that our

ATSDL framework outperforms contemporary models in terms of semantics and syntactic structure and obtains competitive results when performing a manual linguistic quality evaluation.

[12] Nikhil S. Shirwandkar et. Al presented "Extractive Text Summarization Using Deep Learning" to highlight significant portions from the text while preserving the summary's coherence and sense of losslessness. The summaries were produced using English-language text materials. A range of sentence and word level features are used to construct meaningful sentences. Fuzzy logic and the limited Boltzmann machine are used to provide two summaries for each page.

[13] Nabil Alami et. Al researches on topic “Enhancing unsupervised neural networks based text summarization with word embedding and ensemble learning” they uses integrated unsupervised deep neural network techniques with word embedding approach and ensemble learning approach to find summarized data.

[14] Rupal Bhargava et. Al proposed “Deep Extractive Text Summarization” They employed a data-driven methodology and deep learning to produce extractive summaries. The method suggested employs paraphrasing to categorise sentences as candidates for inclusion in summaries or not. The suggested model includes a number of hyperparameters that are customised for the particular data set. Para Multiling 2015 data they uses.

[15] Oliva et. Al developed a model called SyMSS to estimate similarity of sentences the interplay between the systaltic structures of two distinct phrases are examined. The rooted sentence and related sentences may both be found using a syntactic dependency tree. According to these approaches, each word in a phrase has certain syntactic links, which give each sentence its meaning.

[16] Mihalcea et. Al represented Each phrase is represented as a list of word vectors, with the first phrase being the main sentence. They utilised the greatest semantic similarity measurements between the main sentence and the next sentence to determine the word-to-word similarity measure. Throughout this process period, the method will be repeated till the second primary sentence could be located.

[17] Joshi et. Al introduce SummCoder, a framework for the universal extraction of single documents. Sentence encoder, often referred to as vector representation, summary synthesis, and pre-processing are the three components of this system. Removing superfluous whitespace, special characters, URLs, email addresses, and numbers is the initial stage in document validation. The skip-thoughts method is then used to convert each phrase into a fixed-length vector (actual numbers). The summary was effectively constructed using three techniques: sentence novelty metre, sentence + position relevance metric, and sentence-contents relevance metric. This approach produces significantly better results, but it lacks accuracy measurements.

[18] In order to train the extractive model and the abstractive model independently, Yenchun et. Al combined the reinforcement learning technique. The two networks are connected using the gradient approach of sentence-level strategy.

[19] BERT was used by Wang et. Al to encrypt each sentence. They tested three training strategies—Mask, Replace, and Switch—and showed that they were successful.

3. Methodology:

3.1 STRUCTURE OF ATS

One of the different components that make up the fundamental framework of an ATS is an input layer that may receive one or more documents. a single-document summarizing (SDS) method that considers the word count of the summary while selecting the essential terms from the source text [20]. A Multi-document Summarization (MDS), on the other hand, builds the summary utilizing a large number of articles as input [21]. Pre-processing, feature extraction, application of a summary generating technique, or ways to summarize the original material are some of the processes that an automatic text summarizing often goes through. While a multi-document summarization system relies less on the document's structure than a single document summary system, the single document summary system uses the document's monolithic structure. The main issue with MDS is that data is gathered from several sources, some of which could have more redundant information than is typically present in a single document. Additionally, assembling data from several publications into a coherent summary by condensing it into a single sentence. The summarising of news items, academic papers, emails, product reviews, lecture notes, Wikipedia page creation, medical records, and software project activities are a few examples of real-world uses for this type of work [28]. Both extractive and abstractive summarizing techniques exist. The abstractive method is more practical than the extractive method for creating a concise summary of conflicting ideas and preferences. The basic elements and processes of an ATS are visually shown in Figure 2 and are described below:

PRE-PROCESSING

Sentence segmentation and punctuation marks are important strategies used in the pre-processing phase of the process to pre-process input text documents.

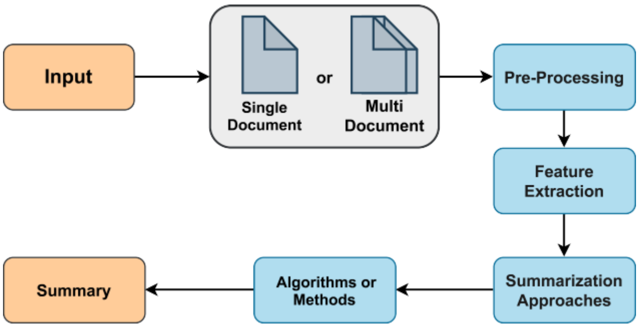

FIGURE 2. Basic elements of an ATS.

The graphic shows seven processes that include pre-processing the input (single or multiple documents) and feature extraction. The next part offers a summary of the paper after approaches and strategies for summarization.

FEATURE EXTRACTION

Sentences must be extracted from the source material using various criteria by choosing aspects that are relevant to the whole summarizing process. Each sentence is given a specific characteristic during the feature extraction stage, and the summary's picked from the terms with the highest scores.

A. SUMMARIZATION APPROACHES

The most crucial and first step in the summarizing strategy is to select efficient techniques. Other strategies involve condensing and paraphrasing a single remark, while others involve selecting the most crucial words and sentences from the texts.

B. ALGORITHMS

The definition of text summarization in terms of algorithms or techniques is more exact. Numerous strategies and methodologies are utilized to generate a more accurate version of the summarized text.

3.2 DATASETS USED FOR AUTOMATIC TEXT SUMMARIZATION

Newspapers, articles or blogs, reviews, emails or communications categorized by genre, and summary datasets are the four main sources used in these datasets. TeMario Corpus, CNN News, and Daily Mail dataset are sources for newspaper data. EASC, LCSTS, and Wikihow are sources for articles and blogs. Opinosis and New Taiwan Weekly are sources for reviews. SKE and the Enron dataset are sources for emails and messages.

3.3 PRE-PROCESSING TECHNIQUES IN ATS

There are several pre-processing stages carried out to quiet the noisy and unfiltered content. "Noisy" and "unfiltered text" are phrases used to characterize inappropriate messages and dialogues, which may include slang or trash language. The following procedures appear to be some of the pre-processing strategies most frequently employed:

1) Parts Of Speech (POS) Tagging: Part-of-speech (POS) tagging is the process of assigning grammatical tags to each word in a sentence. It categorizes words into parts of speech such as nouns, verbs, adjectives, etc. POS tagging can aid in extracting and understanding the role of each word within the text, facilitating subsequent analysis and information extraction.

2) **Stop Word Filtering:** Stop words are common words that do not carry significant meaning for analysis, such as "and," "the," or "in." Removing stop words can reduce noise and improve the efficiency of subsequent text processing steps. A predefined list of stop words can be used, or statistical methods can be employed to identify frequently occurring words as stop words [23].

3) Lemmatization and Stemming: Techniques like lemmatization and stemming are used to break down words to their root or basic forms. Lemmatization maps words to their dictionary forms (lemmas), considering grammatical rules and word context. Stemming, on the other hand, truncates words to their root form using simpler heuristic rules. Both techniques help consolidate different forms of the same word and reduce redundancy in the data. [23].

4) **Named Entity Recognition (NER):** NER is a method for locating and categorising named entities in text, such as names of individuals, businesses, places, and other particular phrases. NER helps in extracting and categorizing relevant information from resumes, such as educational institutions, companies, job titles, and skills, enabling better matching and analysis [24].

5) Tokenization: Tokenization is the process of dividing the text into tokens, which can be words or subwords. It makes it simpler to manipulate and analyse certain text parts. Whitespace splitting, rule-based approaches, or more sophisticated strategies like word embeddings or subword tokenization can all be used for tokenization[25], [26].

6) Capitalization: It could be challenging to utilise various capitalization techniques in various papers, thus It's important to convert all of a text's letters to lowercase letters. Using this technique, all text and documents' words are subsequently amalgamated into a single feature space. [27]

7) Slang and Abbreviation: At the pre-processing step, two distinct forms of text anomalies—slang and abbreviations—are addressed. Support vector machines are examples of acronyms, which are words or phrases that are mostly constructed from the first letters of each term.

8) Noise Removal: Additionally, the bulk of textual data includes additional characters like punctuation and special letters. While special characters and prominent punctuation are necessary for human reading of texts, classification algorithms might get hampered by them [29].

9) Spelling Check: An optional spelling check is part of the pre-processing stage. Typos are regularly encountered in texts and documents, particularly in text datasets from online media (like Twitter) [30].

10) Lemmatization: Lemmatization, often known as a lemma, is the process of getting a word's core word form by either adding a new suffix or eliminating an existing one. Its primary area of use is in natural language processing [31].

3.3 DEEP LEARNING ALGORITHM

Deep learning algorithms play a crucial role in automatic text summarization by leveraging the power of neural networks to understand and generate summaries from text data. Here are some ways in which deep learning algorithms contribute to automatic text summarization. These models are highly promising for ATS since they attempt to imitate how the human brain functions. Deep neural networks are commonly used in NLP applications because to how well their architecture suits the complex language structure. For instance, before passing the results to the layer below, each layer might do a particular duty. Here are a few well-known examples of deep learning-based ATS models:

RNN Encoder-Decoder: The RNN encoder-decoder architecture can be applied to automatic text summarization tasks, including generating abstractive summaries. Here's how an RNN encoder-decoder model can be utilized for automatic text summarization[32]:

1. Encoder: The encoder component of the model processes the input text, typically a document or a long piece of text, such as an article. The encoder, usually implemented using recurrent neural networks like LSTM or GRU, sequentially processes the input text and captures the contextual information of each word or token. The final hidden state of the encoder represents the summarized understanding of the input text.
2. Attention Mechanism: To improve the summarization process, an attention mechanism is commonly incorporated. Attention allows the decoder to focus on different parts of the encoded text during the decoding process. It aligns the encoded information with the relevant parts of the input text, enabling the model to selectively attend to important words or phrases.
3. Decoder: The decoder component takes the encoded representation of the input text and generates the summary. It decodes the information step-by-step, generating words or phrases one at a time based on the encoded information and the attention mechanism. At each decoding step, the decoder considers the previously generated words, the encoded input, and the attention weights to generate the next word in the summary[33].

Long Short-Term Memory (LSTM): A particular sort of recurrent neural network (RNN) architecture is called LSTM (Long Short-Term Memory). that can be employed in Applicant Tracking Systems (ATS) for various tasks. Here's how LSTM can be used in an ATS[6]:

1. Resume Parsing: Resume parsing is a common task in ATS where the system extracts relevant information from resumes and organizes it into structured formats. LSTM can be utilized to process the textual content of resumes and extract key information such as contact details, educational qualifications, work experience, skills, etc. The LSTM model can learn to recognize patterns and dependencies in the resume text, enabling accurate extraction of information.
2. Skill Matching: Matching job requirements with candidate profiles is a crucial task in an ATS. LSTM can be utilized to encode job descriptions and candidate resumes into fixed-length representations, capturing the semantic meaning and context of the text. These encoded representations can then be compared using similarity metrics to identify the level of skill matching between a job description and a candidate's profile.

LSTM's ability to capture sequential dependencies in the text can help in understanding the relevance and importance of specific skills.

3. Resume Summarization: Summarizing lengthy resumes into concise summaries can assist recruiters in quickly evaluating candidate profiles. LSTM-based encoder-decoder architectures can be employed for abstractive summarization. The LSTM encoder encodes the input resume text, capturing its context, and the LSTM decoder generates a summary based on the encoded information. LSTM's sequential processing ability helps in understanding the relationships between different parts of the resume, facilitating the generation of coherent and informative summaries[5].

Gated Recurrent Unit (GRU): Another recurrent neural network (RNN) architectural type that may be utilised in a variety of applications is the GRU (Gated Recurrent Unit), including Applicant Tracking Systems (ATS). GRU shares similarities with LSTM but has a simpler structure with fewer parameters. Here's how GRU can be utilized in an ATS[5]:

1. Sequence Modeling: In an ATS, sequence modeling is often required to process sequential data such as resumes, job descriptions, or cover letters. GRU may be used as a sequence model to identify dependencies and connections between textual phrases or tokens. GRU's gating mechanism allows it to selectively retain or update information based on its relevance, enabling the model to capture long-term dependencies efficiently.

2. Sentiment Analysis: GRU models can also be used for sentiment analysis tasks in an ATS. By training a GRU model on labeled data, it can learn to classify resumes or cover letters into positive, negative, or neutral sentiment categories based on the emotional tone of the text. GRU's ability to capture dependencies and context in sequential data aids in modeling the sentiment patterns present in the resumes.

3. Entity Recognition: Similar to LSTM, GRU can be employed for named entity recognition (NER) tasks in an ATS. By training a GRU model on labeled data, it can learn to identify and classify named entities such as names, organizations, locations, or specific terms present in resumes. GRU's sequential processing capability enables it to capture the contextual information necessary for accurate entity recognition.

Restricted Boltzmann Machine (RBM): A specific kind of generative stochastic artificial neural network known as a Restricted Boltzmann Machine (RBM) belongs to the larger class of Boltzmann Machines. RBMs are used for unsupervised learning tasks such as dimensionality reduction, feature learning, collaborative filtering, and more[34].

Here's an overview of how a Restricted Boltzmann Machine works:

1. Architecture: An RBM consists of two layers: a visible layer and a hidden layer. The visible layer represents the input data, while the hidden layer captures higher-level features. Each neuron in one layer is connected to every neuron in the other layer, but there are no connections within each layer.

2. Energy-Based Model: RBM is an energy-based model, which means it gives each arrangement of visible and hidden units a specific energy value. Based on the RBM's weights and biases, the energy of a certain configuration is determined. Lower energy configurations are considered more likely or desirable.

3. Training: Training an RBM involves maximizing the likelihood of the training data. This is done through a process called Contrastive Divergence (CD) or its variants. CD starts with an input data sample and uses it to compute the activations of the hidden layer neurons. These activations are then used to reconstruct the visible layer. The difference between the reconstructed visible layer and the original input is used to adjust the weights and biases of the RBM, aiming to minimize the reconstruction error[35].

.**Naive Bayesian Classification:** The text's most important keywords are retrieved using the naïve Bayesian classification technique [37]. By analysing the unique keyword properties of a text, the Bayes technique employs machine learning to extract the keyword from the input. To improve summary accuracy, this naive Bayesian, score, and timestamp technique is integrated. [38], It focused on creating text summaries using the basic Bayesian classification approach.

Query Based: Sentences are rated in query-based text summarization according to the frequency counts of words or phrases [39]. Sentences with a search phrase performed better than those with only one search term. [40] primarily use query-based text summarising algorithms to extract, for the output summary, the phrases with the highest scores along with their structural contexts.

Generic Summarization:- Generic summarization refers to the task of automatically generating concise and coherent summaries of text documents, regardless of the specific domain or topic. It aims to capture the most important information and main ideas from the source text and present them in a condensed form[39].

There are two main approaches to generic summarization: extractive and abstractive.

1. Extractive Summarization: In order to construct a summary, extractive summarising entails picking out key phrases or sections from the original material. The sentences are typically chosen based on various features such as sentence relevance, salience, informativeness, and coherence. Statistical and machine learning techniques, such as graph algorithms, clustering, or supervised classification, can be used to determine the importance of sentences and select the most representative ones for the summary.

2. Abstractive Summarization: Abstractive summarization goes beyond selecting and rephrasing sentences from the source text. It entails creating new phrases that, although not necessarily being exact, capture the essence of the original text. Abstractive summarization requires natural language generation techniques and often employs deep learning models, such as recurrent neural networks (RNNs) or transformers, to generate summaries. These models learn to understand and generate human-like language, allowing for more flexible and creative summaries[40].

Q-Network : A Q-network, also known as a Q-learning network or Q-value network, is a type of artificial neural network used in reinforcement learning. It is specifically designed to approximate the Q-value function, which represents the expected cumulative reward for taking a particular action in a given state[42]. When the agent decides to include the candidate phrase in the summary, the output Q-value shows the expected value. Q-network use for text synthesis.

Conclusion & Future Scope:

Despite the fact that text summarization is an old problem, scholars are continuously growing intrigued by it. The performance of text summarization as a whole is only average, nevertheless.

Additionally, the summaries that result are not always accurate. As a result, scientists are working to enhance the present text-summarization techniques.

It is possible to increase the intelligence and performance of ATS by integrating it with other integrated systems. In order to summarise and shorten texts, a significant area of study called automatic text summarization is often employed and integrated into a wide range of applications. In this paper, we offer a methodical examination of the vast issue of ATS in multiple phases, including fundamental theories with historical research backgrounds, dataset inspections, feature extraction structures, and major text summarising techniques.

In the future, we may leverage the NLTK model for automated text summarization to train our deep learning model on more real-time data and obtain greater performance rates.

Reference:

[1] H. P. Luhn, ''The automatic creation of literature abstracts,'' IBM J. Res. Develop., vol. 2, no. 2, pp. 159–165, Apr. 1958.

[2] T. Mikolov, K. Chen, G. Corrado, and J. Dean, ''Efficient estimation of word representations in vector space,'' 2013, arXiv:1301.3781.

[3] Z. S. Harris, ''Distributional structure,'' Word, vol. 10, nos. 2–3, pp. 146–162, 1954.

[4] K. W. Church, ''Word2Vec,'' Nat. Lang. Eng., vol. 23, no. 1, pp. 155–162, 2017.

[5] S. Song, H. Huang, and T. Ruan, ''Abstractive text summarization using LSTM-CNN based deep learning,'' Multimedia Tools Appl., vol. 78, no. 1, pp. 857–875, Jan. 2019.

[6] Q. A. Al-Radaideh and D. Q. Bataineh, ''A hybrid approach for Arabic text summarization using domain knowledge and genetic algorithms,'' Cognit. Comput., vol. 10, no. 4, pp. 651–669, Aug. 2018.

[7] K. Ježek and J. Steinberger, ''Automatic text summarization (the state of the art 2007 and new challenges),'' in Proc. Znalosti. Princeton, NJ, USA: Citeseer, 2008, pp. 1–12.

[8] B. Kitchenham and S. Charters, ''Guidelines for performing systematic literature reviews in software engineering,'' Keele Univ., Durham Univ., Durham, U.K., Tech. Rep. EBSE-2007-01, Version 2.3, 2007.

[9] B. Kitchenham, Procedures for Performing Systematic Reviews, vol. 33. Keele, U.K.: Keele Univ., 2004, pp. 1–26.

[10] Al-Maleh, Molham, and Said Desouki. "Arabic text summarization using deep learning approach." *Journal of Big Data* 7 (2020): 1-17.

[11] Song, S., Huang, H., & Ruan, T. (2019). Abstractive text summarization using LSTM-CNN based deep learning. *Multimedia Tools and Applications*, *78*, 857-875.

[12] Shirwandkar, N. S., & Kulkarni, S. (2018, August). Extractive text summarization using deep learning. In *2018 fourth international conference on computing communication control and automation (ICCUBEA)* (pp. 1-5). IEEE.

[13] Alami, N., Meknassi, M., & En-nahnahi, N. (2019). Enhancing unsupervised neural networks based text summarization with word embedding and ensemble learning. *Expert systems with applications*, *123*, 195-211.

[14] Bhargava, R., & Sharma, Y. (2020). Deep extractive text summarization. *Procedia Computer Science*, *167*, 138-146.

[15] Qiu, Y. (2018). *A Questionnaire Integration System Based on Question Classification and Short Text Semantic Textual Similarity* (Doctoral dissertation, Colorado State University).

[16] Mihalcea, R., & Tarau, P. (2004, July). Textrank: Bringing order into text. In *Proceedings of the 2004 conference on empirical methods in natural language processing* (pp. 404-411).

[17] Joshi, A., Fidalgo, E., Alegre, E., & Fernández-Robles, L. (2019). SummCoder: An unsupervised framework for extractive text summarization based on deep auto-encoders. *Expert Systems with Applications*, *129*, 200-215.

[18] Chen, Y. C., & Bansal, M. (2018). Fast abstractive summarization with reinforce-selected sentence rewriting. *arXiv preprint arXiv:1805.11080*.

[19] Wang, B., & Kuo, C. C. J. (2020). Sbert-wk: A sentence embedding method by dissecting bert-based word models. *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, *28*, 2146-2157.

[20] J.-M. Torres-Moreno, ‘‘Single-document summarization,’’ in *Automatic Text Summarization*. Hoboken, NJ, USA: Wiley, 2014, pp. 53–108.

[21] C. Ma, W. E. Zhang, M. Guo, H. Wang, and Q. Z. Sheng, ‘‘Multi- document summarization via deep learning techniques: A survey,’’ 2020, *arXiv:2011.04843*.

[22] S. Petrov, D. Das, and R. McDonald, ‘‘A universal part-of-speech tagset,’’ 2011, *arXiv:1104.2086*.

[23] J. R. Méndez, E. L. Iglesias, F. Fdez-Riverola, F. Díaz, and J. M. Corchado, ‘‘Tokenising, stemming and stopword removal on anti-spam filtering domain,’’ in *Proc. Conf. Spanish Assoc. Artif. Intell.* Berlin, Germany: Springer, 2005, pp. 449–458.

[24] A. Mansouri, L. S. Affendey, and A. Mamat, ‘‘Named entity recognition approaches,’’ *Int. J. Comput. Sci. Netw. Secur.*, vol. 8, no. 2, pp. 339–344, 2008.

[25] G. Gupta and S. Malhotra, ‘‘Text document tokenization for word frequency count using rapid miner (taking resume as an example),’’ *Int. J. Comput. Appl.*, vol. 975, p. 8887, Jan. 2015.

[26] T. Verma, R. Renu, and D. Gaur, ‘‘Tokenization and filtering process in RapidMiner,’’ *Int. J. Appl. Inf. Syst.*, vol. 7, no. 2, pp. 16–18, Apr. 2014.

[27] K. Darwish, ‘‘Named entity recognition using cross-lingual resources: Arabic as an example,’’ in *Proc. 51st Annu. Meeting Assoc. Comput. Linguistics (Long Papers)*, vol. 1, 2013, pp. 1558–1567.

[28] D. L. Whitney and B. W. Evans, ‘‘Abbreviations for names of rock- forming minerals,’’ *Amer. Mineralog.*, vol. 95, no. 1, pp. 185–187, Jan. 2010.

[29] B. Pahwa, S. Taruna, and N. Kasliwal, ‘‘Sentiment analysis—Strategy for text pre-processing,’’ *Int. J. Comput. Appl.*, vol. 180, no. 34, pp. 15–18, Apr. 2018.

[30] J. Schaback and F. Li, ‘‘Multi-level feature extraction for spelling correction,’’ in *Proc. IJCAI Workshop Anal. Noisy Unstructured Text Data*. Princeton, NJ, USA: Citeseer, 2007, pp. 79–86.

[31] J. Plisson, N. Lavrac, and D. Mladenic, ‘‘A rule based approach to word lemmatization,’’ in *Proc. IS*, vol. 3, 2004, pp. 83–86.

[32] L. Lebanoff, K. Song, and F. Liu, ''Adapting the neural encoder- decoder framework from single to multi-document summarization,'' 2018, *arXiv:1808.06218*.

[33] R. Nallapati, B. Zhou, C. Nogueira dos santos, C. Gulcehre, and B. Xiang, ''Abstractive text summarization using sequence-to-sequence RNNs and beyond,'' 2016, *arXiv:1602.06023*.

[34] M. Tanaka and M. Okutomi, ''A novel inference of a restricted Boltzmann machine,'' in *Proc. 22nd Int. Conf. Pattern Recognit.*, Aug. 2014, pp. 1526–1531

[35] N. Jaitly and G. Hinton, ''Learning a better representation of speech soundwaves using restricted Boltzmann machines,'' in *Proc. IEEE Int. Conf. Acoust., Speech Signal Process. (ICASSP)*, May 2011, pp. 5884–5887.

[36] T. Kuremoto, S. Kimura, K. Kobayashi, and M. Obayashi, ‘‘Time series forecasting using restricted Boltzmann machine,’’ in *Proc. Int. Conf. Intell. Comput.* Berlin, Germany: Springer, 2012, pp. 17–22.

[37] N. Ramanujam and M. Kaliappan, ''An automatic multidocument text summarization approach based on Naïve Bayesian classifier using timestamp strategy,'' *Sci. World J.*, vol. 2016, pp. 1–10, Jan. 2016.

[38] J. L. Neto, A. A. Freitas, and C. A. Kaestner, ''Automatic text summarization using a machine learning approach,'' in *Proc. Brazilian Symp. Artif. Intell.* Berlin, Germany: Springer, 2002, pp. 205–215.

[39] O. Tas and F. Kiyani, ''A survey automatic text summarization, ''*Pressacademia*, vol. 5, no. 1, pp. 205–213, Jun. 2017.

[40] A. Mohamed and S. Rajasekaran, ''Improving query-based summariza- tion using document graphs,'' in *Proc. IEEE Int. Symp. Signal Process. Inf. Technol.*, Aug. 2006, pp. 408–410.

[41] Y. Gong and X. Liu, ''Generic text summarization using relevance measure and latent semantic analysis,'' in *Proc. 24th Annu. Int. ACM SIGIR Conf. Res. Develop. Inf. Retr. (SIGIR)*, 2001, pp. 19–25.

[42] I. Sorokin, A. Seleznev, M. Pavlov, A. Fedorov, and A. Ignateva, ''Deep attention recurrent Q-network,'' 2015, *arXiv:1512.01693*.

BIOGRAPHY

Divya Amade is currently pursuing her M.Tech degree in Computer Science and Engineering from Raipur Institute Of Technology affiliated to Chhattisgarh Swami Vivekanand Technical University, Bhilai, Chhattisgarh, India. She has completed BE in Information Technology from Rungta College of Engineering & Technology, Raipur Chhattisgarh, India in 2006. Her research interest fields are Data Science, Neural Network, and Artificial Intelligence.

Rajeshri Lanjewar is currently working as an Assistant Professor in the Computer Science and Engineering Department at Raipur Institute of Technology, affiliated to Chhattisgarh Swami Vivekanand Technical University, Bhilai, Chhattisgarh, India. He is having 10.9 years of experience in teaching. She has published 3 research papers in SCI, Scopus and other reputed international journals and 4 paper presented in National/International conferences.

Vivek Kumar Sinha is currently working as an HOD in the Computer Science and Engineering Department at Raipur Institute of Technology, affiliated to Chhattisgarh Swami Vivekanand Technical University, Bhilai, Chhattisgarh, India. He is having 14 years of experience in teaching. He is currently Research Scholar at Lovely Professional University Phagwara, Jalandhar, Punjab, India. He has published 21 research papers in SCI, Scopus and other reputed journals and conferences.