ISSN (Online) 2394-2320

International Journal of Engineering Research in Computer Science and Engineering (IJERCSE)

Vol 12, Issue 1, January 2025

A Comparative Analysis of Various Text Summarization Approaches

[1] Er. Rahul Garg, [2] Dr. Naresh Kumar Garg

[1] Research Scholar, Maharaja Ranjit Singh Punjab Technical University, Bathinda
[2] Professor, Department of Computer Science & Engineering, Maharaja Ranjit Singh Punjab Technical University, Bathinda
Corresponding Author Email: [1] rahul_ites@mrsptu.ac.in, [2] naresh_cse@mrsptu.ac.in

Abstract— ***This paper reviews the existing methods and approaches involved in Punjabi text summarization, particularly focusing on the Gurumukhi script. We review various extractive summarization approaches, including Term Frequency-Inverse Document Frequency (TF-IDF), graph-based methods like TextRank, and machine learning models. The aim is to provide a comprehensive overview of existing strategies and highlight their applicability to Punjabi text. Through detailed analysis and comparison, the paper identifies key challenges and identifies potential solutions to enhance summarization quality for texts written in Gurumukhi.***

Keywords— ***NLP; TF-IDF; Summarization; Punjabi; Learning.***

I. INTRODUCTION

Human and computer interaction nowadays is one of the important area of research classified as natural language processing. There are variety of tasks out of which one of the significant tasks within NLP is text summarization, which aims to condense large volumes of text into shorter, meaningful summaries. This task is particularly crucial in the modern digital age, where the vast amount of information available requires efficient tools for information extraction and comprehension. There are several uses for text summary, including social media posts, scholarly papers, news stories, and legal documents. It helps the reader or user to easily understand the main ideas without reading through the entire content. This needs to text summarization to save time and helps in better understanding has made this area of research and in past many authors had proposed many types of models and methods for text summarization.

A. Summarization Methods

Summarizing techniques are divided into two categories as extractive and abstractive summarization.

1. **Extractive Summarization**: In this method a sentence or paragraph is taken as input with the goal of summarizing the input text. This approach relies heavily on identifying the most important parts of the text based on various metrics like sentence frequency, relevance, and position. While extractive summarization ensures that the summary contains accurate information from the original text, it often lacks coherence and readability [1].
2. **Abstractive Summarization**: A new sentence is formed as the summarization of input text. This approach is like how humans summarize information, creating summaries that are more coherent and concise. Abstractive methods use deep learning models, such as sequence-to-sequence (Seq2Seq) models with attention mechanisms, to understand and generate summaries [2].

B. Key Techniques and Models

In the past decade there are multiple models and techniques had been proposed to improve text summarizing performance. These include both contemporary deep learning techniques and conventional techniques like Term Frequency-Inverse Document Frequency (TF-IDF).

1. **Traditional Methods**: TF-IDF is the most used traditional model for the extractive text summarization. They analyze the frequency of words and sentences to determine their importance within the text. These methods are straightforward but may not capture the semantic meaning and context effectively [3].
2. **Deep Learning Approaches**: Deep learning has revolutionized text summarization with the introduction of models like Seq2Seq, Transformers, and Graph-based models. These models leverage large datasets and advanced techniques to generate high-quality summaries. For example, Seq2Seq models employ attention techniques to enable the model to concentrate on pertinent passages of the text while producing summaries [4]. These models are able to produce accurate summaries that are both fluid and cohesive [5–6].

C. Evaluation Metrics

The evaluation of a text summarization model is most important phase to evaluate the performance of the model. An important evaluation parameter called ROUGE (Recall-Oriented Understudy for Gisting assessment)used to measures the amount of n-grams that are overlapped between the generated summary and the reference summaries. METEOR (Metric for Evaluation of Translation with Explicit Ordering), BLEU (Bilingual Evaluation Understudy), and human evaluation are other metrics for assessing readability and coherence [7-8].

Despite significant advancements, text summarization faces several challenges. These include handling diverse text structures, maintaining the coherence and readability of summaries, and generating summaries for low-resource languages [9-10]. Future research is expected to focus on addressing these challenges by developing more robust models and leveraging multilingual datasets. In summary, text summarization is a vital area of NLP with wide-ranging applications. Both extractive and abstractive summarization methods have their advantages and limitations. Advances in deep learning and the development of sophisticated models have significantly improved the quality of automatic summaries [11]. Continued research and innovation are essential to overcome existing challenges and enhance the performance of summarization systems across different languages and domains

II. LITERATURE REVIEW

In [12], the author had proposed a summarization and translation method. One of the paper's intentions was to review the extraction of a regularized approach to machine learning which outcomes in more interpretable models that can be easily trained by cross-lingual tasks. The authors told the story of how, in their previous studies, they had developed a novel method for cross-lingual text summation that was based on the combination of both summarization and translation. Their experiment's first step was to use a few basic summaries and then use them in the text. Summarization and translation are the two techniques employed in achieving this, both of which include extractive and abstractive techniques. The aim of the study was to show that they had made the process of translation of the summaries from the English language to the Kannada language much more efficient through their approach.

In [13], the author has created a model for Bengali text summarizing from a word2vector style. This model relies on a vector for each word in the Bengali sentence to perform the same tasks for different sentences. This was planned as a tool to help the system in its learning and to cover the problem. The main benefit of adopting the mixed approach of extractive and abstractive modeling is the generation of succinct summaries. The system was then tested on a set of news articles, and it was found to have given rise to a successful application of the main key topics, which can be used as a base for NLP tools necessitated for low-resource Indian languages.

In [14], the paper was named Discrete Text Summarization (DeTS) which was a new unsupervised method introduced by the author. This approach assesses allowed deductive reasoning, and identifying disambiguation, and nullification of the grammatical constructs. A domain-specific summarizer proposed by the authors referred to as Discrete Text Summarization (DeTS) is also a new four-point tagging method. However, instead of a single summary, these are in a group of several compact and independent points usually cherished as key points. It comes in its pure form and besides, devoid of any use of human performing the humans only need non-proprietary text analysis. The researchers noted the fact that the tool was tested in a real-life global company and was able to understand the underlying meaning of the words even though there was no such translation. The correctness percentage this tool had reached with the target of the comments was high.

In [15] the author scrutinized the most widely used datasets and methods for automatic text summarization in different languages. They placed a spotlight on the supremacy of English datasets, which make up for 75% of the whole resources whereas the challenges that investigators met with low-resource languages such as Arabic and Hindi were discussed. The analysis highlighted the need for more high-quality multilingual datasets to continue the field's productive work across a range of languages and demonstrated that pre-training models have produced the greatest outcomes in summary measures.

The extractive and abstractive framework for the source code summarization is designed by Sun et al. in [16]. In this system, both extractive and abstraction methods are utilized to merge the proficiency of generating factually accurate and natural-sounding summaries. The extractive module spots the important statements and keywords, and subsequently, the abstractive module generates coherent summaries from these factors. The authors conducted vast experiments in datasets of various programming languages that result in their framework outperforming the latest techniques, as they got high points in BLEU, METEOR, and ROUGE-L, as well as human evaluations.

Further in Table I below the existing methods for the ongoing development of summarization techniques in NLP are discussed.

Table I Existing Text Summarization Models

Ref.	Dataset	Model or Learning Method	Conclusion Result
[17]	Collected Corpus	BERT, BART, T5	T5 model was best suited for generating relevant summaries.
[18]	CNN/Daily Mail, GigaWord, DUC 2004	Pointer-Generator networks, Transformer, BERT	The models performed well on unseen text, with varying success across datasets.

Ref.	Dataset	Model or Learning Method	Conclusion Result
[19]	Amazon reviews, CNN news	Sequence to Sequence, LSTM Bidirectional	Both models showed effectiveness with Amazon reviews and CNN news.
[20]	CNN/Daily Mail, DUC 2004	Contrastive Learning, BART	Contrastive learning improved summary faithfulness and quality.
[21]	Long Text Dataset	Sequence-to-Sequence with Pointer Generator Network	Dynamic windowing improved coherence and summary quality.
[22]	MSMO dataset	Seq2seq with attention, Pointer Generator Network, Pointer Generator with Coverage	PGN with coverage achieved the best ROUGE and BLEU scores.
[23]	Arabic Text Dataset	Sequence-to-Sequence with Attention	Dual encoding reduced repetition and improved summary quality.
[24]	Arabic Summarization Dataset	Deep Transformer-based Language Models (TLMs)	TLMs, specifically the PEAGASUS family, outperformed baseline models.
[25]	GigaWord, DUC corpus	Convolutional Seq2seq Model	Model outperformed state-of-the-art alternatives.
[26]	Gigaword dataset	SUMSUG with Semantic Understanding Graphs	Model showed superior performance with fuller semantic information.

III. STEPS

Punjabi text summarization involves condensing Punjabi text documents into shorter versions while preserving their essential information. The process typically involves several key steps, each crucial for achieving high-quality summaries. Below are the detailed steps involved in Punjabi text summarization:

1. Preprocessing

The first phase, called preprocessing, entails cleaning and getting the material ready for summarization. This action consists of:

- Tokenization: Dividing the text into individual words or phrases.
- Normalization: Lowercasing and eliminating punctuation from text to create a consistent structure.
- Eliminating common words that don't add much to the meaning
- Reducing words to their root forms (e.g., "ਲਿਖਣਾ" to "ਲਿਖ") is known as stemming or lemmatization.

2. Sentence Segmentation

In this step, the text is divided into individual sentences. This segmentation helps in identifying the boundaries of sentences, which is crucial for both extractive and abstractive summarization techniques.

3. Feature Extraction

Feature extraction involves identifying key attributes from the text that will help in summarization. Features may include:

- **TF-IDF**: This approach is used to calculate the score of a word in the document based on its importance.
- **Positional Importance**: Sentences appearing at the beginning or end of paragraphs may hold significant information.
- **Cue Words**: Words like "ਮੁੱਖ", "ਨਤੀਜਾ" which indicate importance.

4. Scoring and Ranking

The retrieved features are used to provide a score to each sentence. Higher scoring sentences are deemed more significant and should be included in the summary. Typical techniques for rating include:

- **Sum of TF-IDF Scores**: Summing the TF-IDF values of all words in a sentence.
- **Graph-based Methods**: Using algorithms like TextRank to rank sentences based on their importance and connectivity.

5. Sentence Selection

The best-ranked sentences are chosen in this stage to create the summary. The required length of the summary determines how many sentences are chosen. For extractive summarization, this might involve directly choosing the highest-scoring sentences.

6. Abstractive Summarization (Optional)

The chosen sentences are used to create new phrases that express the primary ideas of the text when abstractive summarization is applied. This involves:

- **Paraphrasing**: Rewriting sentences in a more concise manner.

ISSN (Online) 2394-2320
International Journal of Engineering Research in Computer Science and Engineering (IJERCSE)
Vol 12, Issue 1, January 2025

- **Synthesis**: Combining information from multiple sentences to form a single coherent sentence.

7. Postprocessing

Postprocessing ensures the summary is coherent and grammatically correct. This step includes:

- **Grammar Checking**: Correcting any grammatical errors.
- **Fluency Enhancement**: Ensuring the summary reads smoothly and logically.

8. Evaluation

The final step involves evaluating the quality of the summary. Common evaluation metrics include:

- **ROUGE Scores**: Measures overlap between the generated summary and reference summaries.
- **Human Evaluation**: Involving native speakers to assess the summary's coherence, relevance, and readability.

Algorithm: Punjabi Text Summarization
Input: Punjabi Text Document T
Output: Summary S
1. Preprocessing
1.1 Tokenization:
T = {t1, t2, ..., tn} where ti represents the i-th token in T
1.2 Normalization:
T' = Normalize(T)
1.3 Stopword Removal:
T'' = RemoveStopwords(T')
1.4 Stemming/Lemmatization:
T''' = Stem(T'')
2. Sentence Segmentation
2.1 Segment Text into Sentences:
Sentences = Segment(T''')
3. Feature Extraction
3.1 Compute Term Frequency (TF) for each term ti in each sentence Sj:
TF(ti, Sj) = (Number of occurrences of ti in Sj) / (Total number of terms in Sj)
3.2 Compute Inverse Document Frequency (IDF) for each term ti:
IDF(ti) = log_e(Total number of sentences / Number of sentences containing ti)
3.3 Compute TF-IDF for each term ti in each sentence Sj:
TF-IDF(ti, Sj) = TF(ti, Sj) * IDF(ti)
3.4 Compute Sentence Score for each sentence Sj:
Score(Sj) = Σ TF-IDF(ti, Sj) for all terms ti in Sj

Algorithm: Punjabi Text Summarization
4. Scoring and Ranking
4.1 Rank Sentences by their Scores:
RankedSentences = Sort(Sentences, by=Score, descending)
5. Sentence Selection
5.1 Select top-k Sentences:
SummarySentences = SelectTopK(RankedSentences, k)
6. Abstractive Summarization (Optional)
6.1 Paraphrase and Synthesize Sentences:
AbstractiveSummary = GenerateAbstractiveSummary(SummarySentences)
7. Postprocessing
7.1 Grammar Checking:
CorrectedSummary = GrammarCheck(AbstractiveSummary)
7.2 Fluency Enhancement:
FinalSummary = EnhanceFluency(CorrectedSummary)
8. Evaluation
8.1 Compute ROUGE Scores:
ROUGE = ComputeROUGEScores(FinalSummary, ReferenceSummaries)
8.2 Human Evaluation (Optional):
HumanScores = HumanEvaluate(FinalSummary)
Return Final Summary

The main application areas of this algorithm include:

1. **News Summarization**: Automatically generate short summaries of Punjabi news articles for quick consumption.
2. **Document Summarization**: Summarize lengthy Punjabi documents, such as legal papers, research articles, or government reports, to quickly convey the key points.
3. **Search Engine Optimization**: Provide concise summaries of Punjabi text for search engines to display in search results, improving user engagement.
4. **Content Management Systems**: In platforms managing large volumes of content in Punjabi, this algorithm can help in generating summaries for faster content browsing.
5. **Text-to-Speech Systems**: In speech-based applications, such as digital assistants or audiobooks in Punjabi, the algorithm can summarize text before reading it out loud.
6. **Education**: Help students and educators by summarizing textbooks, articles, or notes in Punjabi for quicker revision or comprehension.
7. **Social Media and Blogging**: Summarize long posts or articles in Punjabi to create short, engaging snippets for social media platforms

ISSN (Online) 2394-2320
International Journal of Engineering Research in Computer Science and Engineering (IJERCSE)
Vol 12, Issue 1, January 2025

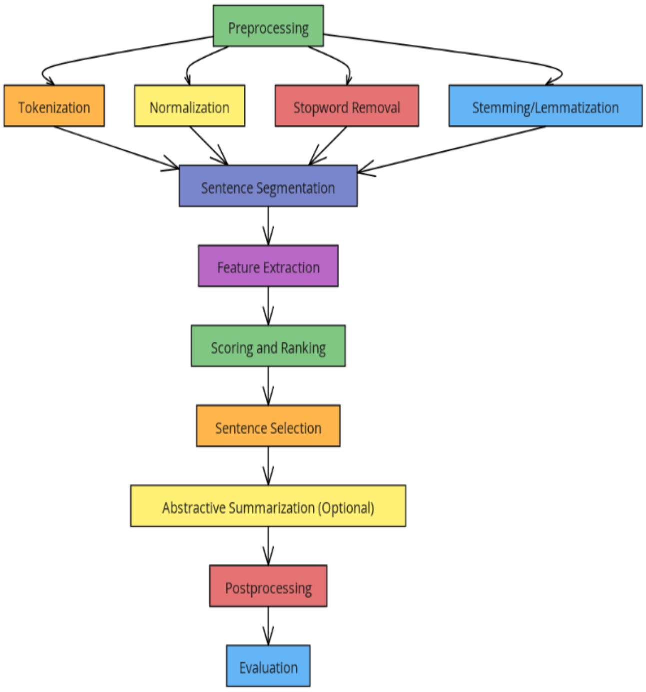

Figure 1. Steps of Text Summarization

IV. SUMMARIZATION METHODS

In order to provide a summary, extractive summarization chooses and takes out important lines from the source material. For extractive summarization, numerous strategies have been developed, ranging from straightforward statistical techniques to more sophisticated machine learning methods. Here are a few typical methods:

1. Term Frequency-Inverse Document Frequency (TF-IDF)

i. The significance of a word within a document in relation to a collection of documents is evaluated using a statistical metric known as TF-IDF.
ii. Method: Compute TF using the given formula. TF(t,d) = Total number of terms in document d / Count of times term t appears in document d.
iii. IDF(t) = log (Total number of documents / Number of documents containing term t) is the formula used to calculate IDF.
iv. Find the TF-IDF score for each sentence and select the top scoring ones.

2. Graph-Based Methods (e.g., TextRank)

Description: Sentences are represented as nodes in a network using graph-based techniques, where the similarity between sentences is shown by edges.

Method:

i. Construct a graph where each node represents a sentence.
ii. Compute edge weights based on sentence similarity (e.g., cosine similarity of TF-IDF vectors).
iii. Apply a ranking algorithm (e.g., PageRank) to score the sentences.
iv. Select top-ranked sentences to form the summary.

3. LexRank

Description: LexRank is a variation of TextRank that focuses on computing sentence importance based on eigenvector centrality in a graph.

Method:

i. Construct a graph similar to TextRank.
ii. Compute cosine similarity to determine edge weights.
iii. Use the degree of centrality to rank sentences.
iv. Select top-ranked sentences.

4. Latent Semantic Analysis (LSA)

Description: Singular value decomposition (SVD) is a method used in LSA to find patterns and connections within the text.

Method:

i. Create a term-document matrix from the text.
ii. Apply SVD to decompose the matrix into singular vectors and singular values.
iii. Use the resulting components to identify key sentences based on their contribution to the main topics.
iv. Select sentences with the highest contribution scores.

5. Centroid-Based Methods

Description: Centroid-based methods identify the central or most representative sentences in a document.

Method:

i. Compute a centroid vector for the document (mean vector of all sentence vectors).
ii. Compute the similarity of each sentence to the centroid.
iii. Select sentences with the highest similarity to the centroid.

6. Machine Learning-Based Methods

Description: These methods use supervised learning to train a model on labeled data (summaries).

Method:

i. Extract features from sentences (e.g., length, position, term frequency).
ii. Train a classifier (e.g., SVM, neural network) on a dataset with labeled summaries.
iii. Use the trained model to score and rank sentences in new documents.
iv. Select top-ranked sentences based on the model's predictions.

7. Reinforcement Learning

Description: Reinforcement learning techniques use an agent that learns to select sentences based on feedback (rewards).

Method:

i. Define a reward function that assesses the chosen sentences' quality.
ii. Train an agent to maximize the reward by selecting optimal sentences.
iii. Use the trained agent to extract sentences from new documents.

8. Maximal Marginal Relevance (MMR)

Description: MMR is a technique that balances relevance and diversity in the selected sentences.

Method:

i. Compute relevance scores for each sentence.
ii. Iteratively select sentences that are both relevant and non-redundant.
iii. Adjust the selection process to maximize marginal relevance.

V. CONCLUSION

In conclusion, the study highlights the importance of developing robust summarization techniques specially for Punjabi texts in the Gurumukhi script. Given the unique linguistic features and the relative scarcity of resources compared to other languages, specialized approaches such as enhanced TF-IDF models, graph-based algorithms, and advanced machine learning techniques show promise. Future work should focus on creating extensive, high-quality datasets and refining these methods to handle the nuances of Punjabi, thereby improving the efficiency and accuracy of automated text summarization in Gurumukhi. This will significantly aid in information extraction and accessibility for Punjabi speakers.

REFERENCES

[1] M. Yadav, "Summarization Of Text Using Natural Language Processing," *International Journal of Scientific and Research Publications*, vol. 13, no. 4, pp. 1-11, 2023. doi: 10.29322/ijsrp.13.04.2023.p13611.

[2] M. Al-Maleh and S. Desouki, "Arabic text summarization using deep learning approach," *Journal of Big Data*, vol. 7, pp. 1-17, 2020. doi: 10.1186/s40537-020-00386-7.

[3] K. Aggarwal, "A Review of Text Summarization Techniques Using NLP," *Computational Intelligence and Machine Learning*, vol. 4, no. 2, pp. 1-15, 2023. doi: 10.36647/ciml/04.02.a001.

[4] Z. Zhou, H. Yu, and G. Fan, "Effective approaches to combining lexical and syntactical information for code summarization," *Software: Practice and Experience*, vol. 50, no. 11, pp. 2313-2336, 2020. doi: 10.1002/spe.2893.

[5] J. M. Karanja and A. Matheka, "A Hybrid Model for Text Summarization Using Natural Language Processing," *Open Journal for Information Technology*, vol. 5, no. 2, pp. 1-10, 2022. doi: 10.32591/coas.ojit.0502.03065k.

[6] L. Huang, W. Chen, Y. Liu, S. Hou, and H. Qu, "Summarization With Self-Aware Context Selecting Mechanism," *IEEE Transactions on Cybernetics*, vol. 52, no. 12, pp. 5828-5841, 2021. doi: 10.1109/TCYB.2020.3042230.

[7] M. R. Desai, B. Gachhinakatti, P. Balaganur, R. Y, and L. Rathod, "Automatic Text Summarization in Natural Language Processing," *2021 IEEE International Conference on Mobile Networks and Wireless Communications (ICMNWC)*, pp. 1-5, 2021. doi: 10.1109/ICMNWC52512.2021.9688499.

[8] A. Bialy, M. A. Gaheen, R. M. ElEraky, A. Elgamal, and A. Ewees, "Single Arabic Document Summarization Using Natural Language Processing Technique," *International Journal of Innovative Technology and Exploring Engineering*, vol. 1, no. 1, pp. 17-37, 2019. doi: 10.1007/978-3-030-34614-0_2.

[9] S. Gorbachev and S. A. Kale, "Extractive Text Summarization using Deep Natural Language Fuzzy Processing," *International Journal of Innovative Technology and Exploring Engineering*, vol. 1, no. 1, pp. 1-10, 2019. doi: 10.35940/ijitee.f1203.0486s419.

[10] D. Maylawati, Y. Kumar, F. Kasmin, and M. Ramdhani, "An idea based on sequential pattern mining and deep learning for text summarization," *Journal of Physics: Conference Series*, vol. 1402, no. 7, pp. 1-10, 2019. doi: 10.1088/1742-6596/1402/7/077013.

[11] W. Sun, C. Fang, Y. Chen, Q. Zhang, G. Tao, T. Han, Y. Ge, Y. You, and B. Luo, "An Extractive-and-Abstractive Framework for Source Code Summarization," *ArXiv*, vol. abs/2206.07245, no. 1, pp. 1-20, 2022. doi: 10.48550/arXiv.2206.07245

[12] Shree Akshaya A T, Shruthi Shankaran, H. M. Thrupthi, Mamatha H R, "Natural Language Processing based Cross Lingual Summarization," *2022 6th International Conference on Trends in Electronics and Informatics (ICOEI)*, pp. 1825-1829, 2022. doi: 10.1109/ICOEI53556.2022.9776655

[13] Sheikh Abujar, Abu Kaisar Mohammad Masum, M. Mohibullah, Ohidujjaman, S. A. Hossain, "An Approach for Bengali Text Summarization using Word2Vector," *2019 10th International Conference on Computing, Communication and Networking Technologies (ICCCNT)*, pp. 1-5, 2019. doi: 10.1109/icccnt45670.2019.8944536

[14] Abbas Golestani, Kristen Robinson, Melissa McMahan, Joe Meyer, "Discrete Text Summarization (DeTS): A Novel Method Grounded in Natural Language Inference and Grammatical Sequence Scoring," *2023 IEEE Ninth International Conference on Big Data Computing Service and Applications (BigDataService)*, pp. 191-196, 2023. doi: 10.1109/BigDataService58306.2023.00041

[15] Alaa Ahmed AL-Banna, Abeer K. Al-Mashhadany, "Natural Language Processing For Automatic text summarization [Datasets] - Survey," *Wasit Journal of Computer and Mathematics Science*, 2022. doi: 10.31185/wjcm.72

[16] Weisong Sun, Chunrong Fang, Yuchen Chen, Quanjun Zhang, Guanhong Tao, Tingxu Han, Yifei Ge, Yudu You, Bin Luo, "An Extractive-and-Abstractive Framework for Source Code Summarization," *ArXiv*, vol. abs/2206.07245, pp. 1-20, 2022. doi: 10.48550/arXiv.2206.07245

[17] Sanskruti Badhe, Mubashshira Hasan, Vidhi Rughwani, Reeta Koshy, "Synopsis Creation for Research Paper using Text Summarization Models," *2023 4th International Conference for Emerging Technology (INCET)*, pp. 1-5, 2023. doi: 10.1109/INCET57972.2023.10170144

[18] Rumisa Lateef, M. Wani, "Performance Comparison of Abstractive Text Summarization Models on Short and Long Text Instances," *2021 8th International Conference on Computing for Sustainable Global Development (INDIACom)*, pp. 125-130, 2021. doi: 10.1109/INDIACom 51348.2021.00023

[19] Chandu Parmar, R. Chaubey, Kirtan Bhatt, Reena Lokare, "Abstractive Text Summarization Using Artificial Intelligence," *SSRN Electronic Journal*, 2019. doi: 10.2139/ SSRN.3370795

[20] Shusheng Xu, Xingxing Zhang, Yi Wu, Furu Wei, "Sequence Level Contrastive Learning for Text Summarization," *Proceedings of the AAAI Conference on Artificial Intelligence*, vol. 36, no. 10, pp. 11556-11565, 2021. doi: 10. 1609/aaai.v36i10.21409

[21] Leon Schüller, Florian Wilhelm, Nico Kreiling, Goran Glavas, "Windowing Models for Abstractive Summarization of Long Texts," *Lecture Notes in Computer Science*, pp. 384-392, 2020. doi: 10.1007/978-3-030-72240-1_39

[22] Md Ashraful Islam Talukder, Sheikh Abujar, Abu Kaisar Mohammad Masum, Sharmin Akter, S. A. Hossain, "Comparative Study on Abstractive Text Summarization," *2020 11th International Conference on Computing, Communication and Networking Technologies (ICCCNT)*, pp. 1-4, 2020. doi: 10.1109/ICCCNT49239.2020.9225657

[23] Kaichun Yao, Libo Zhang, Dawei Du, Tiejian Luo, Lili Tao, Y. Wu, "Dual Encoding for Abstractive Text Summarization," *IEEE Transactions on Cybernetics*, vol. 50, no. 1, pp. 985-996, 2020. doi: 10.1109/TCYB.2018.2876317

[24] Hasna Chouikhi, M. Alsuhaibani, "Deep Transformer Language Models for Arabic Text Summarization: A Comparison Study," *Applied Sciences*, 2022. doi: 10.3390/ app122311944

[25] Yong Zhang, Dan Li, Yuheng Wang, Yang Fang, W. Xiao, "Abstract Text Summarization with a Convolutional Seq2seq Model," *Applied Sciences*, 2019. doi: 10.3390/app9081665

[26] Congcong You, Xiaopeng Cao, Weiwei Zhao, Guansheng Peng, "SUMSUG: Augmented Abstractive Text Summarization Model with Semantic Understanding Graphs," *Proceedings of the 2022 5th International Conference on Artificial Intelligence and Pattern Recognition*, 2022. doi: 10.1145/3573942.3573964