Comparative Study on Extractive Summarization Using Sentence Ranking Algorithm and Text Ranking Algorithm

1st Mansoora Majeed
Dept. of Computer Science And Engineering
Government Engineering College
Thrissur, India
mansooramajeedvv@gmail.com

2st Kala M T
Dept. of Computer Science And Engineering
Government Engineering College
Thrissur, India
kalakrishnakumar22@gmail.com

2023 International Conference on Power, Instrumentation, Control and Computing (PICC) | 979-8-3503-3446-3/23/$31.00 ©2023 IEEE | DOI: 10.1109/PICC57976.2023.10142314

***Abstract*—It is possible to create an accurate and succinct summary using a method called automatic text summarising. Before creating the necessary summary sentences, the machine learning algorithms may be trained to understand documents and recognise the portions that transmit crucial facts and information. Abstractive text summa- rization and Extractive text summarization are its two methods. An extractive text summarization refers to the process of removing significant text from a text document or source manuscript. Deep learning techniques are used to train an abstractive text summarizer, which then generates new phrases and keywords based on the content of the original text. The process of acquiring and assimilation of the knowledge from numerous sources takes time. Automatic text summary is used to speed up the process of finding and consuming pertinent information. An extractive text summarising technique chooses crucial phrases, sentences, and other parts of the original text and concatenates them into a compressed form. The extractive text summarization produces grammatically correct sentences. The project is to develop a model that demonstrates extractive text summarization using sentence ranking and text rank algorithm. It shows that the text rank algorithm does a remarkable performance to extractive text summarization. Using a sentence ranking algorithm, extractive summaries are created that rate sentences according to their weights. In order to create a high-quality summary of the input document and store the summary as audio, the input document's highly scored sentences are extracted. A non-supervised, extractive text summarising method is called TextRank. While calculating the value of words and phrases in a document, the TextRank algorithm looks at how they relate to one another. According on how important the words are in each phrase, it then assigns a score to each one. The summary is formed using the algorithm's choice of the most crucial phrases.**

***Index Terms*—Extractive Text Summarization, Abstractive Text Summarization, Text Rank Algorithm, Sentence Rank Algorithm, Deep Learning.**

I. Introduction

Automatic text summarization aims to shorten lengthy articles because doing it manually may be time-consuming and expensive. Before creating the necessary summary sentences, the machine learning algorithm may be trained to understand documents and recognise the portions that transmit crucial facts and information. As an illustration, a summary was created using a machine learning algorithm that was fed the picture of the news story. [1].

One technique for extracting the key ideas from a paper or group of linked documents and condensing them into a shorter version while maintaining their main points is text summarising. It shortens the time needed to read the entire document. Tools for automatically summarising material that make it simple for users to gain insights are needed. At the moment, there is easy access to a vast amount of information. Long papers are automatically condensed into shorter versions, a process that can be time-consuming and expensive to complete manually. Extraction and abstraction are two alternative methods for automatically summarising material. The approaches for extractive summarization function by locating key passages in the text and creating them. While abstractive summarization techniques aim to produce significant information in a novel way. It might be difficult to come up with grammatically and semantically sound sentences while summarising abstractive texts. The following approach for text summarising is called extractive text summarization. because it is a simpler strategy. [2].

The sentence ranking approach, which gives a concise summary of the input text, is discussed. The weights that are assigned to sentences determine how they are ranked. In order to create a high-quality summary of the input material and store the summary as audio, highly scored sentences are taken from the input text. The text processing algorithm text rank is a graph-based ranking algorithm that is used to locate the most pertinent phrases in a text as well as to find keywords and convert a summary into audio format.

II. RELATED WORK

Before creating the necessary summary sentences, the machine learning algorithm may be trained to understand documents and recognise the portions that transmit crucial facts and contents. In order to produce high-quality summaries and high-quality keywords, which are necessary for text summarising, this work adopts text rank-based automatic

text summarization. Before creating the necessary summary phrases, the machine learning algorithm may be trained to understand documents and recognise parts that transmit crucial facts and information. In this research article [3] text rank-based automatic text summarization is being used to provide high-quality summaries and high-quality keywords, both of which are necessary for text summarization. The effectiveness of the suggested summarising approach is evaluated using the F-measure score.The outcome of research study shows 2 percentage higher outcomes compared to previous work.

The T-bertsum: Topic-aware text summarization based on bert [4] suggests the T-BERTSum topic-aware extractive and abstractive summarization approach, that is based on Transformers' bidirectional encoder representations (BERTs). The suggested technique can simultaneously infer themes and provide summaries from social messages, which is an advance over earlier approaches. To direct the topic generation, the embedded representation of BERT is compared with the encoded topic representation created by the neural topic model (NTM). The transformer network jointly investigate topic inference and text summarization in an end-to-end way after learning the long-term relationships. Third, the effective information is filtered on the abstractive model using a gated network, and the long short-term memory (LSTM) network layers are piled on the extractive model to collect sequence timing information.To communicate the knowledge, a two-stage extractive-abstractive model is also built. In contrast to earlier work, the suggested model T-BERTSum emphasises on topic mining and pretrained external knowledge to capture more accurate contextual representations. When compared to the most sophisticated strategy, experimental findings on the CNN/Daily Mail(DM) and XSum datasets show that our suggested model generates consistent themes while achieving new state-of-the-art outcomes.

In Extractive text summarization using sentence ranking [5],it demonstrates an unique statistical technique for extracting text summarization on a single document. Sentence extraction is a method that presents the concept of the supplied text in a condensed manner. The weights that are assigned to sentences determine how they are ranked. It extracts essential sentences from the input material by using highly scored sentences, which results in a high-quality summary of the input document. These process of automatically extracting the most crucial information from a text is known as text summarising. On a single document, it explains extractive text summarization.

The system that was put out causes a paradigm shift in how neural extractive summarization systems are created. We construct the extractive summarising job as a semantic text matching issue, in which a source document and candidate summaries will be matched in a semantic space, as opposed to the traditionally utilised paradigm of extracting phrases separately and modelling the connection between sentences. Notably, our thorough investigation of the inherent gap between sentence-level and summary-level extractors based on the characteristic of the dataset provides strong support for this paradigm change to a semantic matching framework. [6]

On several NLP tasks, BERT has displayed ground-breaking performance. In this article, we discuss BERTSUM, a straightforward BERT version for extractive summarization. On the CNN/Dailymail dataset, our system outperforms the previous best-performed system by 1.65 on ROUGE-L, making it the state-of-the-art. We need BERT to produce the representation for each sentence in order to utilise it for extractive summarization. The output vectors of BERT, on the other hand, are grounded to tokens rather than sentences because it was trained as a masked-language model. In contrast, extractive summarization uses many labels, whereas BERT only uses two labels (sentence A or sentence B). This is because extractive summarization uses segmentation embeddings to indicate various phrases. As a result, we change BERT's input sequence and embeddings to enable summary extraction. [7]

Author proposed an extractive text summarizer in "Automatic Extractive Text Summarization using K- Means Clustering" [8], in which we give document into preprocessing stage. Lemmatization, stopword elimination, and tokenization are all steps in the preprocessing stage. The feature extraction units receive the preprocessed words after which they create a cosine similarity matrix and perform clustering using the k-means method. The sentence extraction is then given the extracted tokens. then received the summary. Nevertheless, this approach has several drawbacks, such the need to specify the number of clusters in advance. Because the highly rated sentences could be identical, redundancy elimination strategies are needed. Moreover, some phrases may include many topics, but each sentence can only belong to one cluster.

A system that inputs the text and performs stopword removal, word stemming, and part of speech tagging is proposed in "Text Summarization Using Latent Semantic Analysis Model on Mobile Android Platform" [9]. Next an input matrix is created using the Latent Semantic Analysis technique. Singular Value Decomposition is then provided. Following that, sentences are chosen to create summaries. This approach also has certain drawbacks, such as the fact that the resulting summary depends on how well the input is represented semantically and that running SVD takes a lot of time.

It was suggested in "Word Sequence Models for Single Text Summarization" [10] Identifying the most pertinent information in the source document is the biggest challenge for producing an extractive mechanised text summary. Recently, several methods have been effective in identifying the candidate text fragments for the summary by using the word sequence information from the self-text for such a purpose. In this study, we investigate the benefits and drawbacks of extractive text summarization using so-called n-grams and maximum frequent word sequences as features in a vector space model. The phases of term selection, term weighting, sentence weighting, and sentence selection are frequently accomplished using an extractive summarization technique. The technique for the sentence selection stage is, however, restricted to only choosing the strongest sentences. Although

this method may be effective for the first ranked sentence, it may also indicate that subsequent phrases that are identical to the first tend to be ranked after the first, leading to repetitive sentences in the summary. Recall measurement is negatively impacted by this issue. We use the unsupervised learning technique to automatically identify groups of similar sentences from which the best representative sentence is chosen, hence decreasing the repetition in the summary and eliminating this issue.

The goal of the project described in "Extractive Text Summarization using Deep Natural Language Fuzzy Processing" [11] is to condense the text while maintaining the information contained within it. In summary, given complicated tasks that are essentially going to need deep natural language fuzzy processing approaches. Generally speaking, an extractive based summary method is the very straightforward original text of subset of which will not guarantee as best narrative coherence output, because they are most conveniently representing an approximative summarised content from given text-based only on relevance judgement. Pre-processing (sentence segmentation, tokenization, stop words removal), Feature Extraction, Sentence Scoring, Sentence Ranking, and Summary Extraction are the several processes in an artificial fuzzy summarization method.

III. THE PROPOSED SYSTEM

The extractive text summarization using sentence ranking method and text rank algorithm are proposed and to differentiate between Recall and F-score of both these techniques.

A. Design Description of Sentence Ranking Method

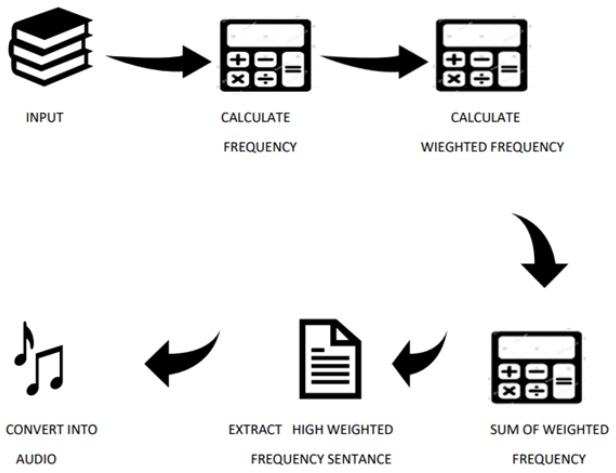

Fig. 1. System Architecture Of Sentence Ranking Method

The design of the sentence ranking method as given in fig. 1, is divided mainly into differernt phases i.e., Reading the given text, data preprocessing ,calculating the weight frequency of tokens, finding the Individual terms ranks and extracted summaries are converted into audio form. In the first phase, we need to input the data. Then the data in the form of text is tokenized as words. Also need to remove the stopwords, punctuations and newline character. Then calculate frequency of each word. Then need to calculate weighted frequency of each token. After getting the weighted frequency, calculate score of each sentence. Then extracts highest ranked sentences and extracted summaries are converted into audio form. These are the modules used in the system's design.

1) Data Preprocessing: Firstly read the given input and the data is given to the preprocessing stage. We are taking input as text. In the preprocessing step,

- The text which is given as input is tokenized to get tokens of the terms.
- After tokenization, the stop words are eliminated from the text. The remaining words are considered as a keyword.
- Then punctuation and newline character is also removed from the text.

2) Calculating the Weight Frequency of Tokens: For calculating the weighted frequency of each token, before need to find the frequency of each word. Frequency means number of times a word will occur. Then find the maximum frequency of the word in the given text. Then calculates weighted frequency as frequency of the word divided by the maximum frequency. Then gets all the weighted frequency of the word in the given text.

3) Calculating the Score of Each Sentence: For calculating the score of each sentence, need to tokenize the text as sentences. Then we calculate the sum of weighted frequencies. The frequencies are connected in this stage in lieu of the matching words in the phrase, and the total of them is discovered. Based on the weighted frequency, the ranks are determined. The sentences are sorted on the basis of their Weighted frequency ranks like highest rank to lowest. The sentences are sorted in descending order.

4) Extracted Summaries Are Converted into Audio Form: After the ranking of each sentences in the text, need to extract top 'N' sentences by using nlargest function in heapq module. Then join function in python joining all the extracted sentences and return the final summary of the text. After it converts into voice form by using gTTS module. The gets the voice version of the extractive text summarization.

B. Design Description of Text Rank Algorithm

The design of the text rank algorithm is shown in fig 2. Steps are, read the given text and tokenized into sentences , generate similarity matrix, rank sentences in similarity matrix, sort the rank and place top sentences based on rank and extracted summaries are changed into audio form. In the first phase, we need to input the data. Then the data in the form of text is tokenized as sentences. Then find similarity function using cosine distance. Then generate similarity matrix among all the words. After getting the similarity matrix, then create sentence similarity graph. Then rank each sentence by using page rank algorithm. Then extracts the important sentences based on the ranking and extracted summaries are converted into audio form.These are the modules used in the system's design.

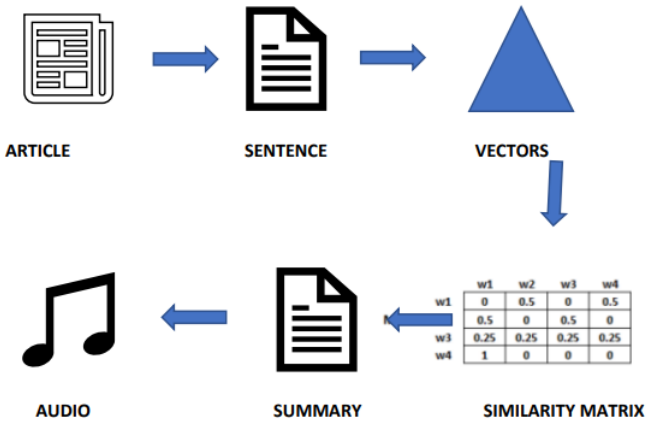

Fig. 2. Architecture Of Text Ranking Method

1) Data Preprocessing: Firstly read the given input and the data is given to the preprocessing stage. We are taking input as text. In the preprocessing step,

- The text which is given as input is tokenized for getting tokens of sentences.
- After tokenization, the stop words are eliminated from the text. The words that are remaining are considered as keywords.

2) Calculate Sentence Similarity and Sentence Score: For calculating the sentence similarity, before need to create vector for the sentences. After creating the vectors, then find the cosine similarity between each and every sentences.

3) Create Similarity Matrix Among All Sentences: For creating the similarity matrix, need to create an empty matrix. Then creating similarity matrix among all the sentences using previously calculated sentence similarity

4) Rank Sentences in Similarity Matrix: After creating the similarity matrix, need to transform into sentence similarity graph. By using the sentence similarity graph, we need to rank the sentences by using page rank algorithm. Then sort it and place top sentences for getting the summary of the given text.

5) Extracted Summaries Are Converted Into Audio Form: After the ranking of each sentences in the text, need to extract top 'N' sentences by using sorted function in python. Then join function in python joining all the extracted sentences and return the final summary of the text. After it converts into voice form by using gTTS module. To gets the voice version of the extractive text summarization.

IV. IMPLEMENTATION

The task is to extract important sentences from the given text and summarized text are converted into voice form. To solve this problem, use two methods for extractive text summarization .That is extractive text summarization using sentence ranking and extractive text summarization using text rank algorithm. First import the required libraries and the data in both methods. Then do the corresponding operations on the data which is specified in methodology.

Hardware Requirements for implementing Extractive Summarization Using Sentence Ranking Algorithm and Text Ranking Algorithm includes Dual core processor, 2 GB RAM, 500 GB Memory. And Software Requirements includes Python(2.7 or 3.3 - 3.6), NLTK, Spacy, Heapq, gTTS, Rouge-Score, en_core_web_sm, Punkt

The data given to the system is in the text format. Our choice is here to want which text is to be summarized and produces the voice as output.

V. EXPERIMENTAL RESULTS

We give input to the model in the text format. After some steps, extract the important sentences and form the summary. Then use gTTS module for converting the extracted summary into the voice form. So the output gets from the systems is in the form of voice. Finally, to evaluate the performance of both the methods of extractive text summarization such as extractive text summarization using sentence ranking and text rank algorithm using F1 measure and recall.

A. Evaluation

Rouge is frequently utilised to assess text summarizations. Rouge-1, Rouge-2, and Rouge-L are utilised here. Rouge-N counts how many n-grams in the system produced text and our model match each other. Bigram is made up of successive words, whereas a unigram consists of just one. The longest common subsequence between our model and the reference text is measured by Rouge-L. Find the recall and f-measure scores using these three.

B. Result Analysis

Table I shows the Rouge value comparison of the sentence ranking method
Table II shows the Rouge value comparison of the text rank algorithm.

TABLE I
TABULAR REPRESENTATION OF SENTENCE RANKING METHOD

	ROUGE-1	ROUGE-2	ROUGE-L
RECALL	0.2429	0.2394	0.2196
F-MEASURE	0.3909	0.3863	0.3533

TABLE II
TABULAR REPRESENTATION OF TEXT RANK ALGORITHM

	ROUGE-1	ROUGE-2	ROUGE-L
RECALL	0.8644	0.8544	0.5186
F-MEASURE	0.9273	0.9168	0.5563

C. Comparison

The classification metrics such as the recall and F1 measures are seems to be higher in the case of text rank algorithm than sentence ranking method. In the case of Rouge1, both the values of recall and f-measure is higher in text rank than sentence ranking method that shows in Figure 3. In Figure 4, Similarly recall and f-measure values in Rouge-2 and Rouge-L

is higher in text rank algorithm than sentence ranking.method that shows in Figure 5. So from the comparison, we can say that text rank algorithm is possess high performance than sentence ranking method.

Fig. 3. Rouge-1 Score

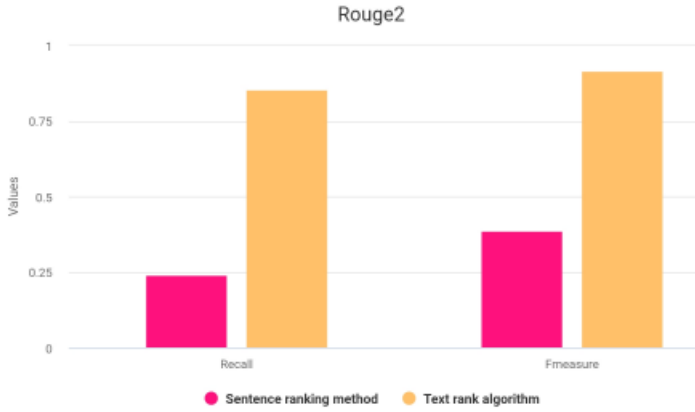

Fig. 4. Rouge-2 Score

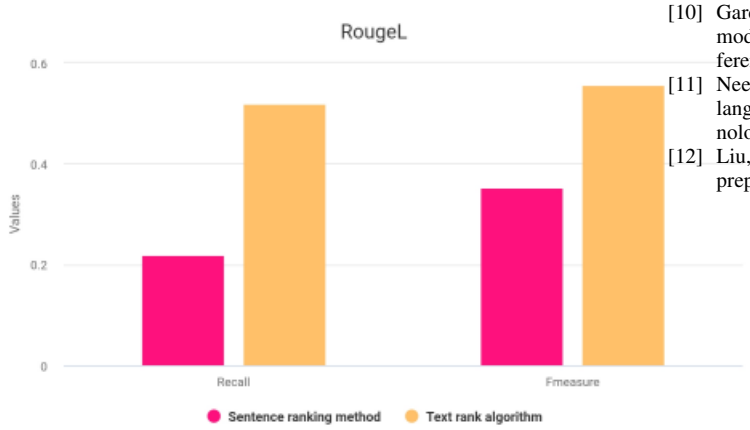

Fig. 5. Rouge-L Score

VI. CONCLUSION

Extractive text summarization which produces the summary of the given text is one crucial Natural Language Processing (NLP) activity is text summarization. By selecting the key words or sentences from the original text and piecing together chunks of the material to create a condensed version, extractive techniques try to summarise articles. The summary is then created using these extracted sentences.

Successfully examined extractive text summarization using sentence ranking method and text rank algorithm. We observed the best results in text rank algorithm. Also the extracted summaries are converted into audio format. The existing extractive text summarization techniques produce the grammatically correct summary. But the sequences of sentences may not be suitable for the smooth reading. So this need to overcome.Extractive summarization should consider semantic part also.

REFERENCES

[1] Mihalcea, Rada, and Paul Tarau. "Textrank: Bringing order into text." Proceedings of the 2004 conference on empirical methods in natural language processing. 2004.

[2] Foong, Oi-Mean, Suet-Peng Yong, and Farha-Am Jaid. "Text summarization using latent semantic analysis model in mobile android platform." 2015 9th Asia Modelling Symposium (AMS). IEEE, 2015.

[3] Yadav, Anurag Kumar, Mukesh Kumar, and Ayonija Pathre. "Implemented Text Rank based Automatic Text Summarization using Keyword Extraction." International Research Journal of Innovations in Engineering and Technology 4.11 (2020): 20.

[4] Ma, Tinghuai, et al. "T-bertsum: Topic-aware text summarization based on bert." IEEE Transactions on Computational Social Systems 9.3 (2021): 879-890..

[5] Madhuri, J. N., and R. Ganesh Kumar. "Extractive text summarization using sentence ranking." 2019 international conference on data science and communication (IconDSC). IEEE, 2019.

[6] Zhong, Ming, et al. "Extractive summarization as text matching." arXiv preprint arXiv:2004.08795 (2020).

[7] Liu, Yang. "Fine-tune BERT for extractive summarization." arXiv preprint arXiv:1903.10318 (2019).

[8] Shetty, Krithi, and Jagadish S. Kallimani. "Automatic extractive text summarization using K-means clustering." 2017 International Conference on Electrical, Electronics, Communication, Computer, and Optimization Techniques (ICEECCOT). IEEE, 2017.

[9] Foong, Oi-Mean, Suet-Peng Yong, and Farha-Am Jaid. "Text summarization using latent semantic analysis model in mobile android platform." 2015 9th Asia Modelling Symposium (AMS). IEEE, 2015.

[10] García-Hernández, René Arnulfo, and Yulia Ledeneva. "Word sequence models for single text summarization." 2009 Second International Conferences on Advances in Computer-Human Interactions. IEEE, 2009.

[11] Neelima, G., et al. "Extractive text summarization using deep natural language fuzzy processing." International Journal of Innovative Technology and Exploring Engineering (IJITEE) 8.6S4 (2019).

[12] Liu, Yang. "Fine-tune BERT for extractive summarization." arXiv preprint arXiv:1903.10318 (2019).