2023 International Conference on Advances in Computing, Communication and Applied Informatics (ACCAI) | 979-8-3503-1590-5/23/$31.00 ©2023 IEEE | DOI: 10.1109/ACCAI58221.2023.10201008

Implementation of Novel Test Rank Algorithm for Effective Text Summarization

Dr. D.Ganesh[1], Mr. Mungara Kiran Kumar[2], Ms. Jasti Varsha[3], Mr.Kimavath Jayanth Naik[3], Ms.K.Pranusha[3] and Ms.Javvaji Mallika[3]

[1]*Associate Professor of CSE, Mohan Babu University (Erstwhile Sree Vidyanikethan Engineering College(Autonomous), Tirupati, Andhra Pradesh , India*
[2]*Senior System Architect, BlueRole Technologies Pvt. Ltd, Hyderabad*
[3]*UG Scholar , Department of CSE, Sree Vidyanikethan Engineering College(Autonomous), Tirupati, Andhra Pradesh , India*

E-mail : ganesh.d@vidyanikethan.edu kirann.intell@gmail.com jastivarsha72@gmail.com jayanth99naik@gmail.com pranushakaduru@gmail.com javvajimallika248@gmail.com

Abstract- Despite their busy schedule, people often wish to know the recent updates of what`s going on around with-in a short span of time. We are often amazed to know what could be the technology behind this technological innovation. Here comes into light Text Encapsulation. To accomplish the task of encapsulating the text into "Manually Tag Articles" or "Design an Automated System for phrase extraction". As the amount of unstructured text data in the digital world grows, automatic summarizing systems are needed that will be possible for people to quickly understand the source material. We require an extractive NLP model that is capable of doing the task on a wide scale. We use text ranking algorithm for text extraction where the number of common words measure's the similarity between two sentences. It gives the most informative document or summary which is also used to find the most relevant sentences in the text.

Keywords Text Encapsulation, Text Summarization, Extractive NLP, Text Extraction, Text Rank Algorithm

I.INTRODUCTION

Summarizing text is the process of extracting the most essential information from a longer piece of writing. It is a method of compressing and extracting data from input documents. This technique is the most important part of Natural Language Processing (NLP) and information retrieval [1]. It is the process of reducing the number of sentences from a large document without changing its actual meaning. Before producing the necessary summarized language, It is possible to teach ML algorithms to comprehend manuscripts and identify the passages that contain crucial information. Text summarization approaches can be categorized into abstractive [2] and extractive synthesis [3]. Using the extractive text summarizing method, key phrases from the source text are taken and combined to create a summary. Without altering the words, the extraction is carried out using the specified measure. The paraphrasing and condensing of portions of the original text are part of the abstraction process. The abstractive text summarization algorithms produce new words and phrases that, like human beings, convey the most important information from the source text. For this reason, generalization is preferable than retrieval.

The text categorization algorithms required for abstraction are more difficult to develop, thus extraction is still widely utilized. The text summary created by this approach is divided into two categories: indicative and instructive. The indicator summary highlights the document's subjects and is mostly used for rapid categorization [4]. These issues are elaborated in an informative summary, which includes conclusions, ideas, and recommendations based on the reader's interest [5]. Any extractive summarization contains three key tasks: intermediate representation, scoring sentences, and summary sentence selection. Figure 1.1 depicts the process of extractive text summarization.

In this scenario, an extractive single-document summarizing method based on graphs is used. Text Rank [6] displays document sentences as nodes in a network with undirected edges and weights depending on sentence similarity. To determine which phrase to comprise in the synopsis, the centrality of a node is frequently estimated by means of graph-based ranking algorithms such as Page Rank [7]. This text score is frequently used for keyword extraction, automatic text summarization, and phrase ranking. This method is used to analyze citations. In fact, Text Rank was influenced by Page

Rank, which is largely used to rank web pages in online search results. Instead of web pages, we use sentences, and the similarity of any 02 sentences is identical to chance of a web page change. To construct a graphical representation of the similarity among phrases in the document, the similar are stored in a square matrix, analogous to the composite M used for Page Rank.

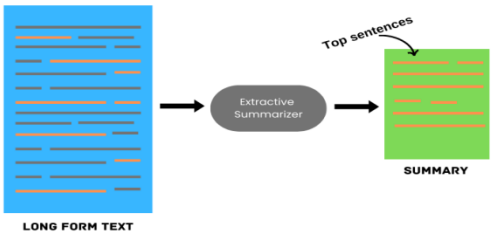

Fig 1.1 Text Summarization

II.LITERATURE SURVEY

First In 2000 U. Hahn and I. Mani [11]proposed that Summarization is the abstracting key content from sources. They investigated for different summarization tools and some knowledge rich and knowledge poor approaches to automatically extract the information which are major challenges of automatic text summarizers. In this paper they have found that knowledge rich approaches can grasp the meaning of text and provide a better summary. An approach to text categorization based on the grouping and extraction of sentences was proposed by Cun-he-li et al.[12] in 2009. Similarity between sentences is determined by first grouping them into groups based on their semantic distance from one another within the document, and then performing a similarity calculation on each group. and finally based on topic sentences extraction of summary. This paper discussed that extraction of sentences for summary will be based on even sentence similarity not only on sentence features.

One of the most important NLP application is text summarization. A novel approach was proposed by M. Aref et.al [13]in 2012 that to generate abstractive review using a semantic graph reducing technique. According to this technique from the input document a semantic graph will be generated and then by using reduced graph an abstractive summary will be generated. From this approach the original text will be reduced upto fifty percent. Cheng-Ying-Liu et.al[14] in 2015, to summarize SNS stream comments there was a problem of incremental clustering . In this paper IncreSTS algorithm was used The clustering results can be updated in real time to reflect the most recent comments, and a quick-glance visualization interface was developed for that purpose.

In 2016 Narendra Andhala and L.A bewoor [15] proposed the importance and meaning of text summarization .They also have discussed on various techniques for Text summary that is automated. Automatic text summarization is becoming an essential method for accurately discovering useful information in massive texts in a short time with minimum effort. This work presents a thorough examination of both extractive and abstractive summarizing approaches.

Yogesh kumar et al[16]. proposed in march 2016, focuses on extractive summarization approaches. Important sentences are chosen for extractive summarization based on various characteristics. The goal of this study is to grasp the semantics of text by using Manage fuzziness and approximate values with the use of fuzzy logic & word net synonyms. This approach generates three summaries by combining the fuzzy logic method, the bushy route method, and the word net synonyms method. The final summary is produced by picking one of the three summaries based on sentence location. Sameer Sonawane et.al[17] proposed in December 2017, which focuses on the text summarization of news articles. In this paper they proposed a technique which focuses on the difficulty of recognizing most significant sentences and produce coherent summary. The method they've adopted doesn't call for a thorough analysis of the text's meaning; rather, they've relied on how topics develop in the text, which is inferred through lexical chains, and by using word net thesaurus they produced an efficient summary. The limitations they faced overcame by using pronoun resolution and still working to develop more precise methods of ranking the articles, and use them to your advantage.

In 2017 Ying Zhong, Zhuo Tang et.al[18] ,The technique employed is a blend of standard summary generation algorithms and deep learning-based abstract generation algorithms. Multiple documents will be taken and transformed to a single document in this approach, and the abstract view of the document will be obtained. To begin, an enhanced LDA model was used to cluster the sentences across

all papers. Finally, the document will be fed into a tensor flow to obtain the abstract. In the field of Natural Language Processing, one strategy for summarizing texts has been developed in this paper by Cengiz HARK et.al[19] in 2018. In this method texts is preprocessed and transferred the proposed diagram in the structure of expression. Different feature extraction is applied on the charts. This text summarizing model chooses phrases based on linear weighting of significance. It does not necessitate the application of extensive linguistic understanding or the adaptation of this work to various languages.

III. METHODOLOGY:

Extraction and abstraction are just two of the many strategies available for summarizing texts

3.1 Extractive Summarization:

This approach involves selecting key phrases & sentences from the unique text to form the abstract. These selected phrases and sentences are often merged to offer a synopsis of the original text. Extractive summarization can be performed using techniques such as keyword extraction and sentence scoring. The overall architecture is as shown in Figure 2.

3.2 Data Preprocessing:

Data pre-processing is the process of cleaning and preparing text data for use in downstream NLP activities in natural language processing (NLP).[25].This may involve a variety of tasks such

as removing noise and other unwanted characters or mark-up, splitting the text into individual tokens, removing common words that do not add meaning to the text, reducing words to their base forms, labeling each token with its part of speech, and identifying and labeling named entities in the text. The specific steps taken in the data pre-processing process will depend on the needs of the task at hand.

3.3 Sentence Tokenization:

Using a procedure called "sentence tokenization," a lengthy paragraph or document can be broken down into its component sentences. Because many natural language processing (NLP) activities rely on this, including part-of-speech tagging, this is a crucial first stepnamed entity recognition, operate at the sentence level.There are several approaches to sentence tokenization, including rule-based methods and machine learning-based methods. Rule-based methods involve defining a set of rules for identifying the boundaries between sentences, such as looking for punctuation marks like periods and exclamation points. Machine learning-based methods involve training a model on a large dataset of sentence-separated text and using the model to identify sentence boundaries in new text. Sentence tokenization can be a challenging task because there are many variations in the way that sentences are written and punctuated, and some sentences may contain multiple clauses or be written in a way that does not follow standard grammatical rules . As a result, it is common to use a combination of rule-based and machine learning-based approaches to achieve good performance on sentence tokenization tasks.

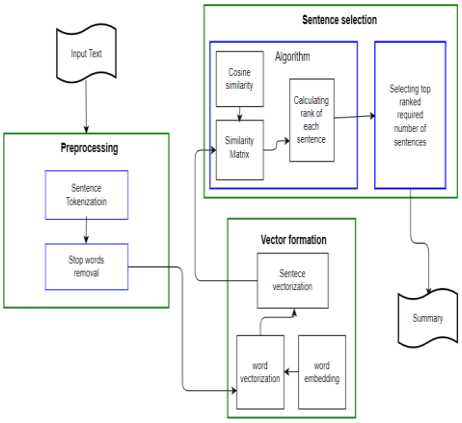

Figure 2.Proposed Text Summarization Procedure

3.4 Stop words Removal:

Stop words are common words that do not add meaningful content to a text and are usually removed as part of the pre processing step in natural language processing (NLP). Examples of stop words include words such as "a," "an," "the," "and," and "but." Removing stop words is important in NLP because these words do not add much meaning to the text and can often add noise to the data, making it more difficult to extract meaningful insights from the text.

There are several approaches to stop words removal, including the use of a pre-defined list of stop words or a machine learning model trained to identify and remove stop words. It is also possible to define custom lists of stop words for specific domains or languages, as the list of stop words can vary depending on the task at hand. It is important to note that stop words removal is not always necessary, and whether or not to remove stop words will depend on

the specific goals and needs of the NLP task at hand. In some cases, stop words may contain important information and removing them may not be desirable.

3.5 Vector Formation:

In natural language processing (NLP), it is often useful to represent text data as numerical vectors in order to apply machine learning algorithms. This process of converting text data into numerical vectors is known as vectorization. The term "sentence vectorization" refers to the procedure of translating a string of words into a numeric vector form. Since many machine learning algorithms only work with numerical data and cannot interpret raw text, this is often helpful in natural language processing (NLP) jobs. Sentence vectorization can be accomplished using a number of methods, such as:

3.5.1 Word Embedding:

Word embeddings are a type of representation for natural language processing (NLP) tasks It represents each word in a lexicon as a dense, continuous-valued vector. Word embeddings capture the meaning of a word in a numerical form and allow words with similar meanings to have similar embedding vectors. Word embeddings can be created in a variety of ways, such as through the use of count-based techniques, predictive techniques, or a combination of the two. Count-based methods create word embeddings by counting the occurrences of words in a corpus and using the counts to create a numerical representation. Predictive methods use a machine learning model to forecast a word given its context, and the model's parameters are used as the word embedding vectors. Hybrid methods combine count-based and predictive approaches. Word embeddings are useful for a wide range of natural language processing tasks, including language modeling, machine translation, and text classification. They can also be used as input to other machine learning models, such as deep neural networks. Using word embeddings allows these models to process text data more effectively, as they are able to capture the meaning of words and their relationships to other words in the vocabulary.

3.6 Sentence Selection:

3.6.1 Text Rank Algorithm:

Text Rank is an algorithm that extracts the most essential sentences or phrases from a text. It is based on the principle of determining the value of each phrase using the text itself rather than depending on external information or established categories. In order to function, Text Rank first models the literature as a structure, where phrases are nodes and the connections between them represent the degree of similarity. The Page Rank algorithm, a technique for rating the significance of nodes in the graph based on the quantity and relevance of the nodes that connect to them, is then applied to each sentence to determine its significance.

3.6.2 Tokenization: The algorithm used by Text Rank begins by breaking the material down into sentences. Several methods exist for this, including separating the text at common punctuation positions like commas and periods .

3.6.3 Similarity computation: The next thing to do is calculate the degree of similarity between every possible pair of sentences. Several methods exist for this, including cosine and Jacquard similarity[27]. When comparing the similarity of two sentences, it is common practice to look for overlapping sets of words and phrases.

3.6.4 Cosine Similarity:

When comparing two non-zero vector in an internal product space, cosine similarity is a useful metric to use because it quantifies how close the vectors are to one another in terms of the cosine of the arc that separates them. Natural language processing & information retrieval rely heavily on it to determine how similar two documents are, or how similar a document is to a query. When two variables A and B are cosine-similar to one another, their dot product is split by the sum of their magnitudes.

$$\text{Cosine similarity} = A \cdot B / \text{sqrt}((A \cdot A) * (B \cdot B)) \quad (1)$$

Where A & B are indeed the vector under consideration, and "." is the sum of two of the vectors. Similarity between two vectors can be expressed as their dot product, whereas individual vector magnitudes are expressed by the sum of squares of the linear combination of the vector amongst itself. The magnitudes of two vectors denote their lengths, and the sum of two of those vectors provides a measure of their similarity. For two vectors, the cosine similarity can take on values between -1 and 1, with an of 1 indicating that the vectors are identical, 0 indicating that they are perpendicular (perpendicular), and a value of -1 suggesting that they are diametrically opposed.

Cosine similarity is widely used as a metric for evaluating the degree of similarity between articles in a collections or between a page and a query. It's especially helpful for comparing documents of varying lengths because it considers the whole length of each vector.

3.6.5 Graph construction: Each phrase is a node in the graph, and the connections between them show the degree to which they are similar to other phrases in the text.

3.6.6 Page Rank: The network is then fed into the Page Rank algorithm to rank the importance of each statement. Over time, the Page Rank algorithm adjusts the weights of individual nodes based on the weights of the networks that link to them. More significant weight is given to nodes that are connected to by more significant nodes.

3.7 Text Rank Algorithm:
Steps to be followed after pre-processing the input text

Input: Document with large amount of data
Output: Summary with reduced number of Lines by removing unwanted data as per algorithm

1 Removal of stop words in the sentences
2 Using word embeddings convert the sentences into vectors.
3 Create a similarity matrix
4 Converting the similarity matrix into a graph in order to use the Page Rank algorithm.
5 Using the Page Rank algorithm to discover the best-ranked phrases.
6 Creating a summary from the top N sentences depending on their rank

IV. Prediction and Evaluation:

The purpose of assessment is to evaluate the results of system-generated and human-generated summaries. This may be accomplished using the metrics cosine similarity, accuracy, recall, and f-score. These metrics are implemented in the system with the help of Python modules.

4.1 Cosine Similarity:
The evaluation of equality constitutes the bulk of the computing load in document execution of tasks, and similarity measure is among the most well-known similarity techniques. According to equation 2, the amount of overlap in words between two texts can be inferred from the cosine similarity between them.

$$Cos(X,Y) = X*Y \backslash | x|*|y| \text{ ----------(2)}$$

The similarity measure value is zero if text X and text Y contain no common words nonetheless, the cosine similarity value may still be determined semantically because the terms are semantically connected.

4.2 Precision:
Precision is used to determine the accuracy of document requested and system generated summary. It is calculated using the formula:

$$Precision=TP/ TP+TF \text{ ----------(3)}$$

4.3 Recall:
Those papers that were recalled and answered the user's system inquiry are used in the recall process. The formula for determining a recall score is as follows:

$$Recall=TP /TP+FN \text{ ----------(4)}$$

4.4 F-score:
The F-score takes into account both the number of correct answers and the number of correct predictions, or recall and precision, respectively. The formula for its determination is as follows:

$$F\text{-score}=Precision * Recall /Precision+ Recall \text{ ---(5)}$$

4.5 ROUGE:
Here we present a set of measures for measuring the accuracy of summaries of written material. The ROUGE metrics compute a score based on the number of n-grams (words or sentences) shared by an automatic summary and one or even more reference summaries. ROUGE-N and ROUGE-L, which measure n-gram overlap and longest common subsequence, respectively, are the most popular ROUGE metrics. The evaluated results is as shown in the figure 2

Table 2.Evaluated Results using Text Rank Algorithm

	Recall	**Precision**	**F-score**
ROUGE-1	0.2352	0.9557	0.3776
ROUGE-2	0.1812	0.9415	0.3039
ROUGE-3	0.2352	0.9557	0.3776

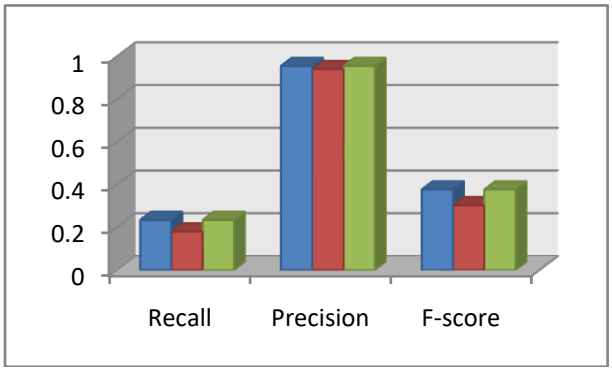

Figure 3.Evaluated Results of the Text Rank Algorithm

V. CONCLUSION:

In this paper, we present a powerful structure and additional topic anchoring information to assist contextual information extraction using an extractive and abstractive approach of text summarization. More information that should have been in the original text can be embedded using a mix of tokens embedding, segment embedding, location embedding, and topic embedding. By stacking the transformers layer in the encoding step, we may improve the system's capacity to store source texts, exploit self-attention, and evaluate the relative relevance of sentence parts using focus scores. For a good summary, an accurate representation is extremely important. We have considered the proposed architecture and implemented the text rank algorithm and shown the evaluated results.

REFERENCES:

[1] Janjanam, P., & Reddy, C. P. (2019, February). Text summarization: an essential study. In 2019 International Conference on Computational Intelligence in Data Science (ICCIDS) (pp. 1-6). IEEE.

[2] Khan, Atif, and Naomie Salim. "A review on abstractive summarization methods." Journal of Theoretical and Applied Information Technology 59, no. 1 (2019): 64-72.

[3] Shirwandkar, Nikhil S., and Samidha Kulkarni. "Extractive text summarization using deep learning." In 2018 fourth international conference on computing communication control and automation (ICCUBEA), pp. 1-5. IEEE, 2018.

[4] Kan, Min-Yen, Kathleen R. McKeown, and Judith L. Klavans. "Applying natural language generation to indicative summarization." arXiv preprint cs/0107019 (2010).

[5] Saggion, Horacio, and Guy Lapalme. "Generating indicative-informative summaries with sumum." Computational linguistics 28, no. 4 (2012): 497-526.

[6] Mihalcea, Rada, and Paul Tarau. "Textrank: Bringing order into text." In Proceedings of the 2004 conference on empirical methods in natural language processing, pp. 404-411. 2014.

[7] Chen, Peng, Huafeng Xie, Sergei Maslov, and Sidney Redner. "Finding scientific gems with Google's PageRank algorithm." Journal of informetrics 1, no. 1 (2017): 8-15.

[8] Manjari, K. Usha, Syed Rousha, Dasi Sumanth, and J. Sirisha Devi. "Extractive Text Summarization from Web pages using Selenium and TF-IDF algorithm." In 2020 4th International Conference on Trends in Electronics and Informatics (ICOEI)(48184), pp. 648-652. IEEE, 2020.

[9] Camacho-Collados, Jose, and Mohammad Taher Pilehvar. "From word to sense embeddings: A survey on vector representations of meaning." Journal of Artificial Intelligence Research 63 (2019): 743-788.

[10] Sohangir, Sahar, and Dingding Wang. "Improved sqrt-cosine similarity measurement." Journal of Big Data 4, no. 1 (2017): 1-13.

[11] Hahn, Udo, and Inderjeet Mani. "The challenges of automatic summarization." Computer 33, no. 11 (2000): 29-36.

[12] Zhang, Pei-ying, and Cun-he Li. "Automatic text summarization based on sentences clustering and extraction." In 2009 2nd IEEE international conference on computer science and information technology, pp. 167-170. IEEE, 2009.

[13] Moawad, Ibrahim F., and Mostafa Aref. "Semantic graph reduction approach for abstractive text summarization." In 2019 Seventh International Conference on Computer Engineering & Systems (ICCES), pp. 132-138. IEEE, 2019.

[14] Liu, Cheng-Ying, Ming-Syan Chen, and Chi-Yao Tseng. "Incrests: Towards real-time incremental short text summarization on comment streams from social network services." IEEE Transactions on Knowledge and Data Engineering 27, no. 11 (2019): 2986-3000.

[15] Andhale, Narendra, and Laxmi A. Bewoor. "An overview of text summarization techniques." In 2020 international conference on computing communication control and automation (ICCUBEA), pp. 1-7. IEEE, 2020.

[16] Jafari, Mehdi, Jing Wang, Yongrui Qin, Mehdi Gheisari, Amir Shahab Shahabi, and Xiaohui Tao. "Automatic text summarization using fuzzy inference." In 2016 22nd International Conference on Automation and Computing (ICAC), pp. 256-260. IEEE, 2016.

[17] Sethi, Prakhar, Sameer Sonawane, Saumitra Khanwalker, and Ravindra B. Keskar. "Automatic text summarization of news articles." In 2017 International Conference on Big Data, IoT and Data Science (BID), pp. 23-29. IEEE, 2017.

[18] Zhong, Ying, Zhuo Tang, Xiaofei Ding, Li Zhu, Yuquan Le, Kenli Li, and Keqin Li. "An improved LDA multi-document summarization model based on TensorFlow." In 2017 IEEE 29th International Conference on Tools with Artificial Intelligence (ICTAI), pp. 255-259. IEEE, 2017.

[19] Cengiz, H. A. R. K., Taner Uckan, Ebubekir Seyyarer, and Ali Karci. "Graph-based suggestion for text summarization." In 2018 International Conference on Artificial Intelligence and Data Processing (IDAP), pp. 1-6. Ieee, 2018.