Automatic Text Summarization

Jay Sharma,[1] Harsh Hardel,[1] Chirag Sahuji,[1] and Rajesh Prasad[1]

[1]Department of Computer Science and Engineering, School Of Engineering,MIT Art, Design and Technology University, Pune, Maharashtra – 412201

---***---

Abstract - *The amount of data on the internet is increasing day by day. So it becomes more difficult to retrieve important points from large documents without reading the whole document. Manually extracting key components from large documents requires a lot of time and energy. Therefore an automatic summarizer is must needed which will produce the summary of a document on its own. In this, a text is given to the computer and the computer using algorithms produces a summary of the text as output. In this paper, we have discussed our attempt to prepare an extraction-based automatic text summarizer in which paragraphs of documents are split into sentences and then these sentences are ranked based on some features of summarization where higher rank sentences are found to be important which is used to generate the summary.*

1. INTRODUCTION

The World of text is vast and is evolving constantly. Most information available on the internet is still in a text format. We humans as frequent users of the internet have to go through large and many different documents to gain that information which takes lots of our time as well as energy. Due to this massive increase in text information, a summary of this information is must needed which will extract the important points from a large set of information or documents. Therefore, a process of producing a concise summary while maintaining important information and the overall meaning of the text is called Text Summarization.

When it comes to summarizing text, humans understand the context of documents and create a summary that conveys the same meaning as the original text but when it comes to automation this task becomes way too complex. Text Summarization is one of the use cases of Natural Language Processing (NLP). Natural Language Processing is the branch of Artificial Intelligence concerned with making computers understand text and words in the same way as human beings used to understand them. We can divide Text summarization into two categories:-

1. **Extractive Summarization**: In this, the most important parts of sentences are selected from the text and a summary is generated from it.

2. **Abstractive Summarization**: In this method, summary involves the words which are generally not present in the actual text it means that it produces a summary in a new way by selecting words on semantic analysis just like how humans read articles and then write the summary of it in their own words.

1.1 Contribution

Over the years, there has been a lot of discussion regarding text summarization and how to deal with its problems. Several papers have been published that discuss various approaches to dealing with automating text summarization. In this paper, we have extended this area in novel ways and contributed to the area of text summarization especially in the Ex- tractive approach because computers still cannot understand every aspect of the Abstractive approach of text summarization. Our study can be summarized as follows:-

- We have presented the TextRank approach to deal with extractive text summarization.

- Also, we have performed evaluation measures and described them in detail.

- Further, we have made a comparison of our proposed system with existing MS-Word approaches.

1.2 Organizational Structure

The outline of the rest of the paper is as follows: In section II we have presented a literature review of text summarization. Section III describes our pro- posed idea/system of extractive text summarization. Section IV describes the working algorithm we have used in our proposed system. In section V, the result of our proposed system is discussed which involves evaluation as well as performance measures of our system. Section VI includes the future scope of our system and concludes our study.

2. LITERATURE SURVEY

There has been a lot of development in the field of Text Summarizing in the past 9 years. With every advancement comes a new technique with a more optimized approach. Some of these techniques are mentioned below

2.1 Abstractive Text summarization Approaches with Analysis of Evaluation Techniques [1]

The aim is to summarize large documents to extract the key components from the text which makes it very convenient to understand the context and main points of the text. This paper was published in the year 2021 and the author was Abdullah Khilji.

The approach described in the paper uses a baseline machine learning model for summarizing long text. This model takes raw text as input and gives a predicted summary as output. They experiment with both abstractive and extractive-based text summarization techniques. For verifying the model they use BELU, ROGUE, and a textual entailment method.

Dataset being used is Amazon fine foods reviews which consist of half a million reviews collected over 10 years. The methodology being used is Sequence to Sequence and Generative Adversarial. However, the paper is missing details of extractive summarization techniques.

2.2 A Comprehensive Survey on Text Summarization Systems [2]

This paper describes the classification of summarization techniques and also the criteria that are important for the system to generate the summary. This paper was published in 2009 and the author was Saeedeh Gholamrezazadeh.

The extractive approach is described as reusing the portions of the main text which are italics, bold phrases, the first line of each paragraph, special names, etc. For Abstractive approach involves rewriting the original text in a short version by replacing large words with their short alternatives.

Drawbacks of Extractive summarization include inconsistencies, lack of balance, and lack of cohesion. Examples of extractive summarization are Summ-It applet and examples of abstractive summarization are SUMMARIST.

2.3 Extractive Automatic Text Summarization Based on Lexical-Semantic Keywords [3]

This paper focuses on Automatic Text Summarization to get the condensed version of the text. The method mentioned also takes into account the title of the paper, the words mentioned in it, and how they are relevant to the document. However, the summary might not include all the topics mentioned in the input text. This paper was published in the year 2020 by A´ngel Hern´andez-Castan˜eda.

The proposed approach first passes the text to feature generation methods like D2V, LDA, etc. to generate vectors for each sentence. Then it undergoes clustering which measures the proximity among different vectors generated in the previous step. Then LDA is used to get the main sentences above the rest of all sentences which are then used to generate the summary.

This paper explains the whole process in a very precise manner however there is less information on the methods being used in some steps of the proposed flow. No info on term frequency and inverse term frequency is present in the paper.

2.4 Abstractive Text Summarization based on Improved Semantic Graph Approach [4]

This paper talks about the graph-based method for summarizing text. It also tells about how to graph-based method can be used to implement abstractive as well as extractive text summarization. This paper was published in the year 2018 and is authored by Atif Khan.

According to almost all graph-based methods text is considered as a bag of words and for summarization, it uses content similarity measure but it might fail to detect semantically equivalent redundant sentences. The proposed system has two parts one for making the semantic graph and the other part for improving the ranking algorithm based on the weighted graph. Finally, after both parts are executed successfully, an abstractive summary of the text is generated.

The paper has good information on the graph based ranking algorithm and improved versions of it. However, there is no mention of the text rank algorithm. Also, a comparison between the graph-based ranking algorithm and text rank ranking algorithm is absent.

2.5 Text Summarizer Using Abstractive and Extractive Method [5]

In this paper the main motivation is to make computers understand the documents with any extension and how to make it generate the summary. In this method the system uses a combination of both statistical and linguistic analysis. This paper was published in the year 2014 and is authored by Ms. Anusha Pai.

The system introduced in the paper takes text input from user. For summarizing this input it firstly separates the phases, then removes the stop words from the input. After this it performs Statistical and Linguistic analysis to generate the summary. This output is then sent and stored in the database.

The system proposed generates summary according to the input given by the user. This can further be improved by adding synonyms resolution to the model which will treat synonyms words as same. Also multiple documents summarization support can be added.

2.6 Evolutionary Algorithm for Extractive Text Summarization [6]

In this paper a new possibility is introduced, abstractive text summarization might compose of novel sentences, which are not present in the original document. The method introduced uses an unsupervised document summarization method which uses clustering and extracting to generate a summary of the text in the main document. This paper was published in 2009 by Rasim Algulie This method uses Sentence clustering to classify sentences based on similarity measures which classify the sentences in clusters. After this objective function is used to calculate their importance. Along with this Modified Discrete Differential Evolution Algorithm is also mentioned in the paper.

No information is given about the Graph-based approach and also no comparison between the graph based approach and objective function is given. Also, the result is less optimized than the approaches we saw in other papers.

2.7 Automatic Keyword Extraction for Text Summarization: A Survey [7]

This paper talks about current present approaches for summarization. It also talks about Extractive and Abstractive text summarization. This paper was published in the year 2017 and is authored by San- tosh Kumar Bharti.

This paper has divided Automatic Text Summarization into 4 types Simple Statistics, Linguistic, Machine Learning, and Hybrid. These are techniques in which we can implement Automatic Text Summarization. In Simple Statistics, we have Inverse Document Frequency, Relative Frequency Ratio, Term Frequency, etc. In the Linguistic Approach, we have Electronic Dictionary, Tree Tagger, n-Grams, WordNet, etc. In Machine Learning we have SVM, Bagging, HMM, etc. Hybrid is the combination of the previous three mentioned.

This paper contains all the techniques present to the date however since 2017 there has been a lot of improvement in this field. This paper needs to be updated with the latest technologies.

2.8 Automatic Keyword Extraction for Text Summarization in Multi-document e-Newspapers Articles [8]

This paper was published in the year 2017 and is authored by Santosh Kumar Bharti. It takes about implementing Extractive Summarization in everyday life for summarizing e-Newspaper articles.

This paper shows the comparison between TF- IDF, TF-AIDF, NFA, and Proposed methods. It uses F-measure as a parameter to measure the efficiencies of the techniques. It can be seen that the proposed method is more efficient than the other three methods.

The paper only mentioned model implementation in Newspaper articles and not in any other type of articles. Also, only Extractive Summarization is used for this model. This model should be tested on different articles and also using different methodologies.

2.9 Graph-based keyword extraction for single-document summarization [9]

This paper focuses on the approach used for selecting keywords from the text input given by the user. The comparison is done between the supervised and unsupervised approaches to identify keywords. This paper was published in the year 2008 by the writer Marina Litvak.

This approach takes into account some structured document features using the graph-based syntactic representation of the text and web documents which improves the traditional vector-space model. For the supervised approach, a summarized collection of documents is used to train the classification algorithm which induces a keyword identification model. Similarly, for the unsupervised approach, the HITS algorithm is used on the document graphs. This is done under the assumption that the top-ranked nodes are representing the document keywords.

The supervised Classification model provides the best keyword identification accuracy. But a simple degree-based ranking reaches the highest F-measure. Also, the only first iteration of HITS is enough instead of running it till we get convergence,

2.10 Extractive approach for text summarization using graphs [10]

This paper implements the Extractive approach but uses a different approach. It uses two matrices for sentence overlap and edits distance to measure sentence similarity. This paper was published in the year 2021 by the author Kastriot Kadriu.

The proposed model takes a document as input. The document undergoes tokenization after which lemmatization is performed and is checked for dependency parsing. Then the model checks for sentence overlap and edits distance if necessary. After this graph representation is done. Now we can apply different algorithms to generate a summary of the text.

In the paper, the author has taken into account different methods for generating the summary. This includes Pagerank, hits, closeness, betweenness, degree, and clusters. Finally, the summary is generated. The paper also shows a comparison between different methods and how accurate their summary is. We use F-score to measure their effectiveness.

2.11 Implementation and Evaluation of Evolutionary Connectionist Approaches to Automated Text Summarization [11]

This paper implemented and compared the performance of three text summarizations developed using existing summarization systems to achieve connectionism. This paper was published in 2010 by the authors Rajesh Shardanand Prasad and Uday Kulkarni.

Three approaches used in the paper are based on semantic nets, fuzzy logic, and evolutionary programming respectively. The results they got were that the first approach performs better than MS Word, the second approach was resulting in an efficient system and the third approach showed the most promising results in the case of precision and F- measure. The paper has used the DUC 2002 dataset to evaluate summarized results based on precision and F-measure.

Approaches used in this paper focus only on small details related to general summarization rather than developing an entire summarization system and thus are only helpful for research purposes.

2.12 Feature Based Text Summarization [12]

This paper aims at creating a feature-based text summarizer that is applied to different sizes of documents. This paper was published in the year 2012 by author Dr. Rajesh Prasad.

This paper follows the extractive way of summarizing the text and utilizes the combination of nine features to calculate the feature scores of each sentence and rank them according to the scores they get. The higher rank sentences are part of the final summary of the text. They have used different types of documents that require different features to get a summary as a data set for their model. This approach gave better results when compared to MS word in terms of precision, Recall and F-measure in most types of documents.

This paper shows great results for different types of documents but some documents may require features more than this paper has used so further research is needed in this approach.

2.13 Review of Proposed Architectures for Automated Text Summarization [13]

This paper aims to review various architectures which have been proposed for automatic text summarization. This paper was published in the year 2013 by its authors Tejas Yedke, Vishal Jain, and Dr.Rajesh Prasad.

In this paper, different techniques for text summarization are discussed and their advantages and drawbacks are also reviewed. DUC 2002 data set is used to calculate results that which approach per- forms better for which type of document in terms of precision, recall and F-measure. The limitation of this paper is that it doesn't provide the most effective technique for summarization.

2.14 Automatic Extractive Text Summarizer(AETS): Using Genetic Algorithm [14]

This paper aims at developing an extractive text summarizer using the Genetic algorithm. This paper was published in the year 2017 by its authors Alok Rai, Yashashree Patil, Pooja Sulakhe, Gaurav Lal, and Dr.Rajesh Prasad.

The approach this paper follows involves feature extraction, fuzzy logic, and a genetic algorithm to train the machine to produce better results in automatic summarization. This paper defines Genetic algorithms as the search strategies that cop with the population of simultaneously seeking positions. Ac- cording to the input file and compression rate given by the user, the authors resulted in forming a meaningful summary as output text. According to the paper genetic algorithm is a sentence-choice-based technique and gives the best results when text summarization is done.

2.15 A Novel Evolutionary Connectionist Text Summarizer (ECTS) [15]

This paper aims to create an efficient tool that can summarize large documents easily using the evolving connectionist approach. This paper was published in the year 2009 by Rajesh S.Prasad, Dr. U.V Kulkarni, and Jayashree R.Prasad.

In this paper, a novel approach is proposed for part of speech disambiguation using a recurrent neural network, which deals with sequential data. Fifty random different articles were used as data sets for this paper. Authors found the accuracy of ECTS was ranging from 95 to 100 percent which average accuracy of 94 percent when compared to other summarizers.

Though this paper has explained the connectionist approach very well there lies some issues in POS disambiguation and deviations found in ECTS for a couple of sentences.

2.16 Abstractive method of text summarization with sequence to sequence RNNs [16]

This paper was published in the year 2019 and is authored by Abu Kaisar Mohammad Masum. It takes about how bi-directional RNN and LSTM canbe used in the encoding layer along with the attention model in the decoding layer for performing abstractive text summarization on the amazon fine food review dataset.

The focus of this paper is Abstractive Summarization by the use of sequence to sequence RNNs. It performs Data Processing in which we Split Text, Add Contractions, Remove stop words and perform Lemmatization. After verifying if the text is purified, we count the size of the vocabulary and add a word embedding file.

The next step is the addition of special tokens which mark important waypoints in the dataset. After this, an Encoder layer and Decoder Layer are present in LSTM and then the Sequence to Sequencemodel is built.

The model is then trained with data for generating the response summary. Even though the above model performs well for short text it suffers when long text input is given. Another drawback is thatit is currently trained for the English language but no such summarizer is available for other languages.

2.17 Automatic Text Summarization Using Local Scoring and Ranking [17]

This paper was published in the year 2017 and is authored by Diksha Kumar. Instead of building a text summarizer, this paper focuses on improvingthe currently available Automatic Text Summarizer to achieve more coherent and meaningful summaries.The model introduced in the paper uses an automatic feature-based extractive text summarizer to better understand the document and improve its coherence. The summary of the given input is generated based on local scoring and local ranking. Wecan select the top n sentences in the ranking for thesummary. Here n depends on the compression ratio of the summarizer. Feature Extraction can be done based on differentcriteria. It can be done based on the frequency of a word appearing in a sentence by selecting the words occurring the most, based on the length of the sentence by avoiding too short or too long sentences, based on the position of the sentence by giving a high score to the first sentence and less to the second sentence and so on, based on sentences overlapping thetitle or heading which can be considered important and based on similarity of a sentence concerning all other sentences in the document.

2.18 A Genetic Fuzzy Automatic Text Summarizer [18]

This paper was published in the year 2009 and is authored by Daniel Leite. This paper focuses on the fuzzy-based ranking system to select the sentences for performing extractive summarization on the input data set.

The fuzzy knowledge base used in this model was generated by a genetic algorithm. For fitness function, ROGUE in formativeness measures were adopted and a corpus of newswire text is being used along with their human-generated summaries for defining the fuzzy classification rules.

The paper also talks about SuPor-2 features which use a Na¨ıve-Bayes probabilistic classifier to find the relevance of a sentence to be extracted from the dataset to be in the generated summary. It has 11 features that address either the surface or the linguistic factors that interact with one another to find the relevance of a sentence. For future scope, the paper talks about using ideal mutation and crossing rates in the evolution phase for the genetic algorithm. Also, membership functions can be explored for modeling fuzzy sets.

2.19 Enhancing Performance of Deep Learning Based Text Summarizer [19]

This paper was published in the year 2017 and is authored by Maya John. This paper aims to enhance the performance of the currently present deep learning model used for text summarization.

In current deep learning models, the summary sentences form the minority class which is very small when compared to the majority class which leads to inaccuracy in a summary generation. To enhance the performance, data can be resampled before giving it to the deep learning model.

The proposed system is divided into steps for simplicity. These are text preprocessing, feature extraction, resampling, and classification. Inside text preprocessing we have tokenization, stop word removal, stemming, and lemmatization. Along with this several resampling, mechanisms are discussed in the paper which can be used on the data set for improving the classifier performance.

2.20 Cut and Paste Based Text Summarization [20]

This paper was published in the year 2000 and is authored by Hongyan Jing. This paper's text summarization model is based on the examination of human written abstracts for a specific text.

The model extracts text from the input for generating a summary and removes the inessential phrases from the text. The phrases are given as output and then joined together to form coherency. This is done based on a statistically based sentence decomposition program which finds where the phrases of a summary begin in the original text input. This produces an aligned corpus of the summary along with the articles used to make the summary.

The model uses a Corpus of human-written abstracts for analyzing the input. It also uses WordNet for Sentence reduction and Sentence combination. Even though it is very accurate, when we test this model with current models it is very simple and old compared to the ones that are being used currently. A lot of advancement has been done in this field since the time this paper was published.

3. PROPOSED SYSTEM

3.1 Preprocessing

Preprocessing involves the taking text as input from the user in different forms like Wikipedia or other links, simple text box and upload text document, etc. and stop words from the sentences are removed.

3.2 Graph Building

These sentences are represented as nodes with all their properties and edges representing interest (similarity) between two sentences.

3.2 Sentence Ranking Algorithm

The concept of Sentence Ranking tells that a document ranks high in terms of ranking, if high ranking documents are linked to it. Here we have used a Textrank algorithm inspired by the Pagerank algorithm which extracts all sentences from input text, create vectors for all sentences, then the similarity between sentence pairs is calculated and finally, sentences are ranked based on score.

3.3 Summarization

Based on the Similarity Score obtained from the similarity matrix the top N sentences are to be included in the final summary and thus Summary is generated from the original text as the output of the system.

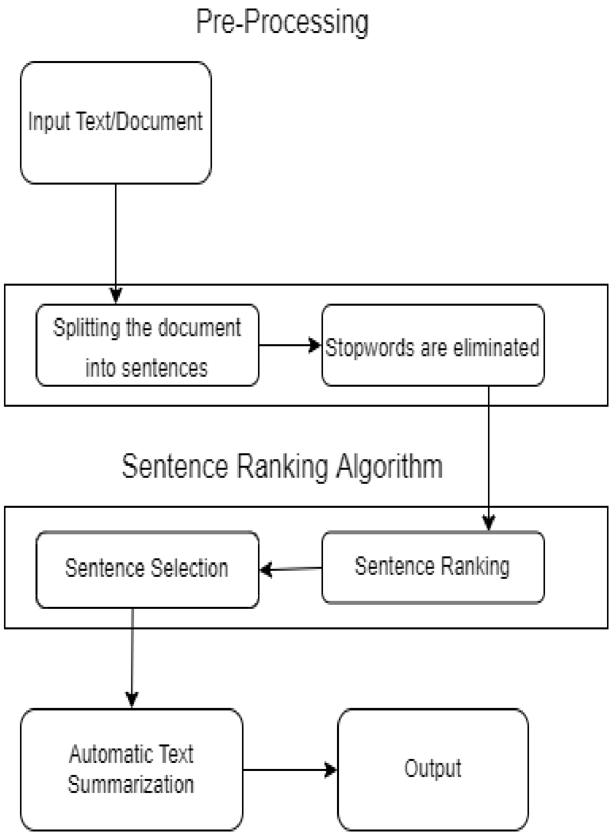


FIG. 1. Proposed System Architecture or Flow Diagram

4. ALGORITHM

4.1 Preprocessing

The algorithm that we have used here is called as Text Rank Algorithm which is inspired by Google's Page rank algorithm and is used for ranking web pages. Consider a web page A which has a link to a web page Z. A PageRank score is calculated based on the probability of users visiting that page to rank these pages. A Matrix is created to store the probabilities of users navigating from one page to another is called Similarity Matrix. The probability of going from page A to Z is M[A][Z] initialized as 1/(number of unique links on the website). Suppose A which contains links to 2 pages, so the contribution of A to PageRank of Z will be PageRank (A)/2. If there is no link between web pages then probability should be initialized as 0. To solve this issue a constant d called as damping constant is added. So final equation will be:

PageRank(Z)=(1-d)+d*(PageRank(A)/2)

Text Rank has similar logic as PageRank but there are some changes like in-place of web pages Text sentences are taken and a similarity matrix is calculated based on similarity values between two sentences using maximum common words. So accordingto the algorithm to create a graph of sentence ranking a vertex of each sentence is created and added to the graph. Further, the similarity between the two sentences is calculated based on the common wordtoken present in the two sentences. In the graph, an edge between two vertices or sentences denotesthe similar content or interest among two sentences. Long sentences are avoided to be recommended in summary by multiplying with a normalizing factor.

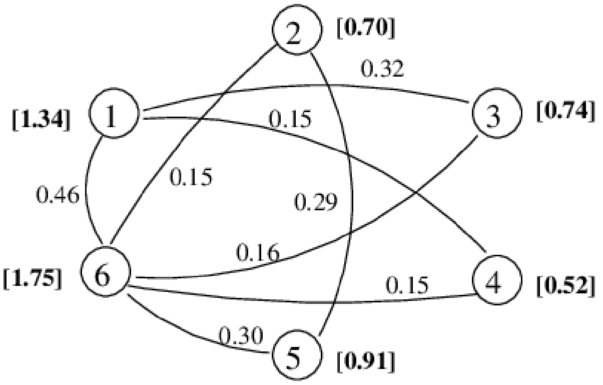


FIG. 2. Similarity graph drawn based on similarity matrix for sentence similarity

The similarity between the two sentences is givenby:

$$Similarity(S_i, S_j) = \frac{|w_k|w_k \epsilon S_i \& w_k \epsilon S_j|}{log(|S_i|) + log(|S_j|)} \quad (4.1)$$

where given two sentences Si and Sj, with a sentence as set N-words appear in that sentence. Let's take an example to illustrate the working of algorithm:

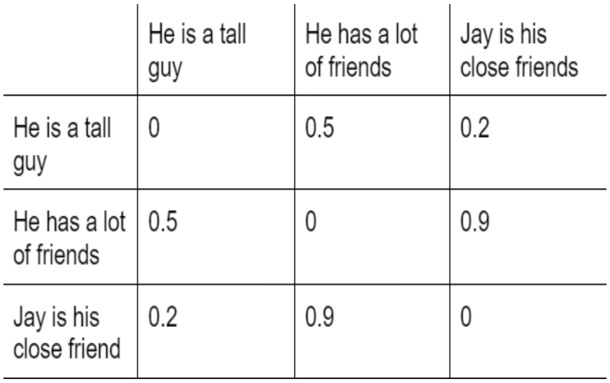
Let there be three sentences as follows:

A=' He is a tall guy '
B=' He has a lot of friends '
C=' Jay is his close friend '
Initialize TextRank(A), TextRank(B), TextRank(C) as 1 and take d as 0.85.

Figure 3 shows how the similarity matrix is created for the above sentences.

TextRank(A)
=(1-0.85) + 0.85 * (TextRank(B) * M[B,A] +

TextRank(C) * M[B,C])
=0.15 + 0.85*(1*0.5+1*0.9) = 1.340
TextRank(B)
= (1-0.85) + 0.85 * (TextRank(B) * M[A,B] + TextRank(C) * M[A,C])
= 0.15 + 0.85*(1.34*0.5+1*0.2) = 0.889
TextRank(C)
= (1-0.85) + 0.85 * (TextRank(A) * M[C,A] + TextRank(B) * M[C,B])
=0.15 + 0.85*(0.889*0.2+1.34*0.9) = 1.326



	He is a tall guy	He has a lot of friends	Jay is his close friends
He is a tall guy	0	0.5	0.2
He has a lot of friends	0.5	0	0.9
Jay is his close friend	0.2	0.9	0

FIG. 3. Similarity Matrix where values shows the similarity between two sentences

The above process is continue for every sentence for n iterations. Therefore all the sentences are arranged according to their text ranks and the most important sentences are added to a summary.
Graphical heat map representation of similarity matrix is shown in FIG. 4

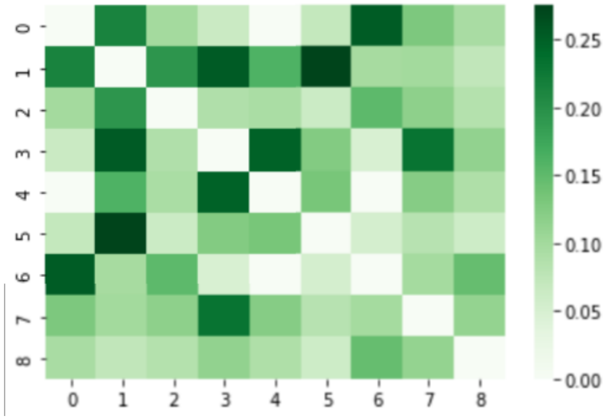


FIG. 4. Graphical representation of Similarity Matrix using a Heat map

4. RESULT AND ANALYSIS

Applying the proposed idea and algorithm we have implemented an automatic text summarization system that takes text as input from the user and gives him precise summary. The experimental results and analysis of our proposed system is been discussed in this section. The proposed summarization system is implemented in Python using Flask and nltk library. For testing purposes, we have used BBC News summary data set from Kaggle. This data set contains the documents of news articles in different categories and their extractive reference summary respectively.

Evaluation Measures: Evaluation of our proposed Text Summarization System is carried out to determine the quality of the summary produced by the system. We have evaluated our system using the ROUGE evaluation measure. ROUGE stands for Recall-Oriented Understudy for Gisting Evaluation which is a set of metrics to evaluate the quality of the summary produced by the system. ROUGE-N measures the number of matching n-grams between our model generated summary and human generated summary. ROUGE-1 measures for uni-gram. Similarly ROUGE-2 measures for bi-grams. ROUGE-L measures the longest common sub-sequence between our model and referenced summary. We have calculated the precision, recall, and f-measure of each ROUGE-1, ROUGE-2 and ROUGE-L. Precision is the proportion of correctness of sentences in the summary. For precision, higher the value, better the system performs to generate summary.

$$\textit{Precision} = \frac{\text{Retrieved Sentences} \cap \text{Relevant Sentences}}{\text{Retrieved Sentences}}$$

The Recall is the ratio of the relevant sentence in the summary.

$$\textit{Recall} = \frac{\text{Retrieved Sentences} \cap \text{Relevant Sentences}}{\text{Relevant Sentences}}$$

F-measure calculates the harmonic mean of precision and recall.

$$F - measure = 2 * \frac{(Precision * Recall)}{(Precision + Recall)}$$

Higher is the value, higher is the similarity between the model generated and referenced summary.

Performance Measures: When it comes to the performance of the summarizer it should give a concise summary of text similar to a human-generated summary. The performance of our system is evaluated on summary available in BBC news dataset using evaluation measures described above. We have taken 5 documents from the dataset of each category. Then we generated a summary for each document using our proposed summarizer. For experimentation, the summary is generated for different percentages of the summary that the user wants to generate, and this generated summary is evaluated on the extractive summary available in the dataset using evaluation measures.

To check for the efficiency of our proposed system we used it to generate summary of different documents and then compared that with the summary generated by one of the currently present system for generating summary. To check the efficiency we have 3 evaluations. First, we compare our proposed system generated summary with the original summary of the input. We can find the original summary of the document in the database along with the input document. By this we can check the accuracy of our model. Secondly, we compare our proposed system generated summary with the summary generated for the same document by some other currently present summarization model. By this we can check the performance of our model with respect to the currently present model. Lastly, we check our system generated summary with the summary made by a human. This is one of the most important evaluation as it gives us the exact idea of the summary that is required by a user and how much of it is present in the summary generated by our model. We have tested our model for 2 different lengths of generated summary: 15% and 30% length of the input document. When the summary for different documents is evaluated at the rate of 15% of the summary the computed evaluation measures we got can be seen in FIG.5 and similarly for a rate of 30% of summary the computed evaluation can be seen in FIG.6.

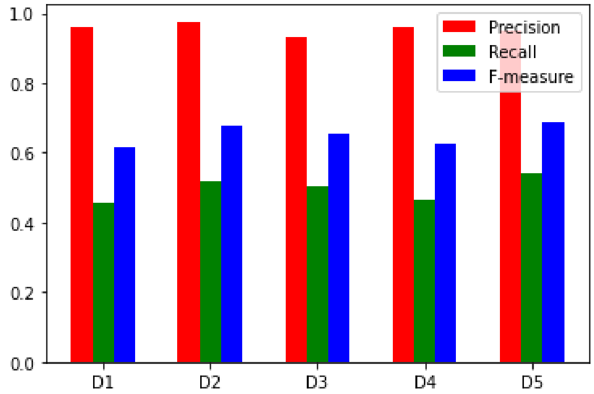


FIG. 5. Comparison of precision, recall and f-measure of 5 documents when summary generated is 15% of the total document length

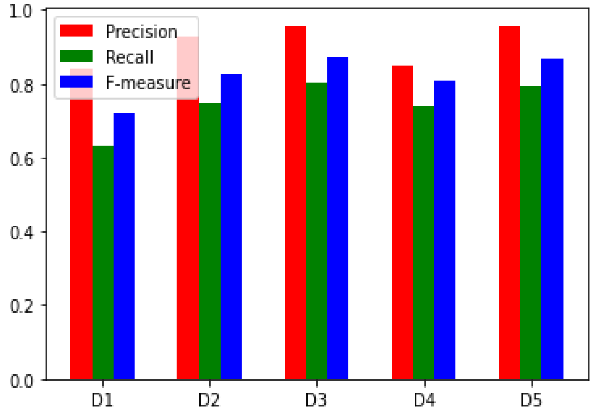


FIG. 6. Comparison of precision, recall and f-measure of 5 documents when summary generated is 30% of the total document length

To test the summarizer we have summarized the different documents from the dataset based on domains of politics, entertainment, sports, technology, etc. This is done to see how our proposed system reacts to documents of different types, different lengths, etc. and how the generated summary is affected by these. Precision, recall, and F-measure for each document was calculated to test the content understanding of our proposed summarization system.

TABLE I. Comparison of precision values of proposed system with respect to existing MS-word system

Methods	Dataset	R-1	R-2	R-L
TextRank	D1	0.9607	0.8833	0.9411
	D2	0.9756	0.9313	0.9756
	D3	0.9324	0.9052	0.9324
	D4	0.9583	0.8852	0.9583
	D5	0.9487	0.9189	0.9487
	Average	0.9551	0.9047	0.9512
MS-Word	D1	0.9534	0.8333	0.9534
	D2	0.909	0.8645	0.909
	D3	0.9	0.8809	0.9
	D4	0.7619	0.5925	0.7619
	D5	0.7391	0.6144	0.7391
	Average	0.8526	0.7571	0.8526

These evaluation measures are further compared with MS WORD 2007 Summarizer systems to test how well our proposed system works compared to the existing summarization tool. The result shown in Table I shows the comparison of average values of precision for different ROUGE measures

i.e. ROUGE-1(R-1), ROUGE-2(R-2) and ROUGE- L (R-L) of our proposed system and MS Word 2007 summarization system, similarly result shown in Table II shows the average values of recall of our proposed system and MS Word 2007 summarization system and Table III result shows the average values of F-measure of our proposed system and MS Word 2007 summarization system.

TABLE II. Comparison of recall values of proposed system with respect to existing MS-word system

Methods	Dataset	R-1	R-2	R-L
TextRank	D1	0.4537	0.3812	0.4444
	D2	0.5194	0.426	0.5194
	D3	0.5036	0.4257	0.5036
	D4	0.4646	0.3802	0.4646
	D5	0.5401	0.4766	0.5401
	Average	0.4963	0.4179	0.4944
MS-Word	D1	0.3796	0.2877	0.3796
	D2	0.4545	0.3721	0.4545
	D3	0.4598	0.3663	0.4598
	D4	0.4848	0.338	0.4848
	D5	0.3722	0.2383	0.3722
	Average	0.4662	0.3736	0.4652

TABLE III. Comparison of f-measure values of proposed system with respect to existing MS-word system

Methods	Dataset	R-1	R-2	R-L
TextRank	D1	0.6163	0.5326	0.6037
	D2	0.6779	0.5846	0.6779
	D3	0.654	0.5791	0.654
	D4	0.6258	0.532	0.6258
	D5	0.5401	0.4766	0.5401
	Average	0.6228	0.541	0.6203
MS-Word	D1	0.543	0.4278	0.543
	D2	0.606	0.5203	0.606
	D3	0.6086	0.5174	0.6086
	D4	0.5925	0.4304	0.5925
	D5	0.3722	0.2383	0.3722
	Average	0.5872	0.4891	0.5858

4. CONCLUSION AND FUTURE WORK

Graphs above show that the Precision, recall and, f-measure values of our model for different length of summary shows that the proposed system extracts a good amount of retrieved sentences from the document in the summary which shows the sign of good accuracy of our proposed summarization system. Also for 30% summary generated we can see that precision, recall and f-measure values are better compared to when 15% summary is generated.

Thus, our system performs better for longer length of summary.

As shown in Table I it is clear that the average precision values of our proposed system are better compared to the values we get from the MS Word summarizer for three documents and are close to equal for two documents. Similarly in Table II and Table III shows that our system also gives better recall and f-measure average values when compared to MS-Word 2007 summarization system.

A Multilingual feature is also added to our summarization model which can help generate summary of different spoken languages text documents so that the people around the world can also use our Text summarization system.

It is concluded that the achieved results of our proposed text summarization model are better in most of the aspects compared to existing models and therefore it can be a good start to further studies.

As extractive summarization is evolving day by day with increase in its research and more algorithms coming out, in the future we will try to get more accuracy in generating summary using extractive approach which would be more feature based i. e it will include more features than just sentence ranking based on correlation between them such as sentence length, word based similarity, title features, etc. Also we would try to use more advanced algorithms like neural networks for generating summary.

The extractive approach is more straightforward because copying big sections of text from the original document ensures grammar and accuracy. Para- phrasing, Generalization and, Assimilation are a part of abstractive summarization. Even though abstractive summarization is a more difficult process, thanks to recent improvements in the deep learning field, there has been some progress. In future it might be possible that the summary generated using extractive approach can be more accurate and meaningful compared to abstractive approach.

5. REFERENCES

[1] Abdullah Faiz Ur Rahman Khilji, Utkarsh Sinha, Pintu Singh, Adnan Ali, and Partha Pakray. Abstractive text summarization approaches with analysis of evaluation techniques. In International Conference on Computational Intelligence in Communications and Business Analytics, pages 243–258. Springer, 2021.

[2] Saeedeh Gholamrezazadeh, Mohsen Amini Salehi, and Bahareh Gholamzadeh. A comprehensive survey on text summarization systems. In 2009 2nd International Conference on Computer Science and its Applications, pages 1–6. IEEE, 2009.

[3] A´ngel Hern´andez-Castan˜eda, Ren´e Arnulfo Garc´ıa Hern´andez, Yulia Ledeneva, and Christian Eduardo Mill´an-Hernandez. Extractive automatic text summarization based on lexical-semantic keywords. IEEE Access, 8:49896–49907, 2020.

[4] Atif Khan, Naomie Salim, Haleem Farman, Murad Khan, Bilal Jan, Awais Ahmad, Imran Ahmed, and Anand Paul. Abstractive text summarization based on improved semantic graph approach. International Journal of Parallel Programming, 46(5):992–1016, 2018.

[5] A Pai. Text summarizer using abstractive and extractive method. International Journal of Engineering Research & Technology, 3(5):22780181, 2014.

[6] Rasim Alguliev, Ramiz Aliguliyev, et al. Evolutionary algorithm for extractive text summarization. Intelligent Information Management, 1(02):128, 2009.

[7] Santosh Kumar Bharti and Korra Sathya Babu. Automatic keyword extraction for text summarization: A survey. arXiv preprint arXiv:1704.03242, 2017.

[8] Santosh Kumar Bharti, Korra Sathya Babu, Anima Pradhan, S Devi, TE Priya, E Orhorhoro, O Orhorhoro, V Atumah, E Baruah, P Konwar, et al. Automatic keyword extraction for text summarization in multi-document e-newspapers articles. European Journal of Advances in Engineering and Technology, 4(6):410–427, 2017.

[9] Marina Litvak and Mark Last. Graph-based keyword extraction for single-document summarization. In Coling 2008: Proceedings of the work- shop Multisource Multilingual Information Extraction and Summarization, pages 17–24, 2008.

[10] Kastriot Kadriu and Milenko Obradovic. Extractive approach for text summarisation using graphs. arXiv preprint arXiv:2106.10955, 2021.

[11] Rajesh Shardan and Uday Kulkarni. Implementation and evaluation of evolutionary connectionist approaches to automated text summarization. 2010.

[12] Rajesh Shardanand Prasad, Nitish Milind Uplavikar, Sanket Shantilalsa Wakhare, VY Jain, Tejas Avinash, et al. Feature based text summarization. International journal of advances in computing and information researches, 1, 2012.

[13] Tejas Yedke, Vishal Jain, and RS Prasad. Review of proposed architectures for automated text summarization. In Proceedings of International Conference on Advances in Computing, pages 155–161. Springer, 2013.

[14] Alok Rai, Yashashree Patil, Pooja Sulakhe, Gaurav Lal, and Rajesh S Prasad. Automatic extractive text summarizer (aets): Using genetic algorithm. no, 3:2824–2833, 2017.

[15] Rajesh S Prasad, UV Kulkarni, and Jayashree R Prasad. A novel evolutionary connectionist text summarizer (ects). In 2009 3rd International Conference on Anti-counterfeiting, Security, and Identification in Communication, pages 606–610. IEEE, 2009.

[16] Abu Kaisar Mohammad Masum, Sheikh Abujar, Md Ashraful Islam Talukder, AKM Shahariar Azad Rabby, and Syed Akhter Hossain. Abstractive method of text summarization with sequence to sequence rnns. In 2019 10th International Conference on Computing, Communication and Networking Technologies (ICCCNT), pages 1–5. IEEE, 2019.

[17] Hari Disle Sameer Gorule Ketan Gotaranr Diksha Kumar, Sumeet Bhalekar. Automatic text summarization using local scoring and ranking. In 2017 international conference on computing methodologies and communication (ICCMC), pages 59–64. IJRTI, 2019.

[18] Daniel Leite and L Rino. A genetic fuzzy automatic text summarizer. Csbc2009. Inf. Ufrgs. Br, 2007:779–788, 2009.

[19] Maya John and JS Jayasudha. Enhancing performance of deep learning based text summarizer. Int. J. Appl. Eng. Res, 12(24):15986–15993, 2017.

[20] Hongyan Jing and Kathleen McKeown. Cut and paste based text summarization. In 1st Meeting of the North American Chapter of the Association for Computational Linguistics, 2000.