DOI: http://dx.doi.org/10.26483/ijarcs.v11i4.6639

Volume 11, No. 4, July-August 2020

ISSN No. 0976-5697

International Journal of Advanced Research in Computer Science

RESEARCH PAPER

Available Online at www.ijarcs.info

TEXT SUMMARIZER FOR URL AND .DOCX FILES

Jasmine Pinto
Computer Engineering
St. Francis Institute of Technology
Mumbai, India

Disha Rathod
Computer Engineering
St. Francis Institute of Technology
Mumbai, India

Alston Quadros
Computer Engineering
St. Francis Institute of Technology
Mumbai, India

Abstract: The objective of this model is to compute a block of text and return its equivalent summary. The text provided could be both content from an online website or an offline text document. The purpose of this project is to help generate an efficient algorithm which could provide a meaningful summary of the text and help the user save time on reading and help grasp maximum information in a minimalistic amount of time.

Keywords: NLP, Text Rank, Word Frequency, Cosine Similarity, POS tagging, Extractive Summarization.

I. INTRODUCTION

With the steady advancements in the field of technology, the internet's growth has also increased to a great extent. In today's fast moving world, information is in great abundance. Information is easily available in the form of text documents, files, statistics and data. As the internet contains more than the required amount of information, it becomes very difficult to search for valuable information through a vast quantity of documents. People need to take decisions and even need to acquire knowledge quickly and on the go. Nowadays, even 24 hours are not enough to complete our day to day tasks and having to read pages and pages of emails, books or even websites would be impossible. To help aid the situation, we have designed an application that will help generate a summary or a brief version of the text provided to the application. Our application mainly emphasizes taking transcripts or data from the user and giving a summary of the transcript or data. User can either enter his own text or enter the URL from where he wants the data to be summarized. On the basis of the data received, we provide an appropriate summary for the same. This will be very beneficial as it will help users to get relevant information from a large pool of data. The user need not read the entire document but just the summary generated by our application to know the content present in the data. It will help users to get a brief idea of the entire document based on the summary generated by our application. The summary provided by our application ensures to include the most relevant information such that the user does not miss on any important points. This will help save on time and gain maximum information from the data.

II. LITERATURE SURVEY

In this paper, extractive text summarization is used to generate the summary. Two summaries are generated for each document using Restricted Boltzmann Machine and Fuzzy Logic. The text document received is pre-processed and sentence features are calculated to find sentence scores. Once sentence features have been calculated for all sentences, a sentence feature matrix is formed. In this matrix, each sentence has nine feature values. The sentence matrix is then normalized and is given as an input to the Restricted Boltzmann Machine to enhance the values for obtaining the sentence score. An enhanced feature matrix is then calculated. For the first summary, the sum of all enhanced feature values is calculated and sentences are arranged in descending manner of their sentence scores. The summary generated will include the first sentence and then top 50% of the remaining sentences based on their descending scores and sorted according to their original position in the document. For the second summary, the feature scores calculated earlier are converted into percentage and triangular membership functions are used to fuzzify each score into high, medium and low levels. Defuzzification is done to determine whether the sentence is Important, Average or Unimportant. The sentences which fit in "Important" category are added in the second summary according to their original position in the document. Both the summaries are combined and a set of common and uncommon sentences are found. The common set is added into the final summary and the uncommon sentences are sorted position wise and half parts of uncommon set of sentences are added in the final summary. After adding, the sentences are arranged according to their original position in the text document. The results obtained show that the proposed system helps to overcome the problem of text overloading by generating an effective summary. They concluded that the results obtained by using the proposed method gives better evaluation parameters in comparison with prevailing RBM method. [1]

In this paper, they have discussed the features and methods for extractive text summarization. They have listed the features that should be applied to exclusive sentences to be included in the summary. They have listed the various supervised and unsupervised learning methods along with their concept, advantages and disadvantages to generate a summary. Various

datasets such as TIPSTER, TREC, TAC, DUC, CNN can be used for experimental evaluation of extractive summarization. After analyzing all the methods for extractive summarization, they came to a conclusion that evaluating summaries is a difficult task due to the impossibility of building a standard against which the results of the system can be compared. They also concluded that their work can be further improved by focusing on the various challenges of extractive text summarization process in premises of time and space compilation. [2]

In this paper, they presented an extensive text summarization approach using neural networks. The neural network had been trained by extracting ten features including word vector embedding from the training dataset. Testing had been performed on the DUC 2002 dataset, where up to 284 documents were used in various test experiments. ROUGE scores (1, 2, and L) computed for their proposed model and four of the online text summarizers showed the effectiveness of the proposed model. They also concluded that the performance of the proposed model may further be improved by increasing the size and diversity of the training dataset and applying more effective approaches to convert the abstract summaries into extractive summaries. [3]

In this paper, a POS tagging method is proposed based on statistical machine learning and SWJTU segmentation dictionary and the method optimizes the POS tagging results of SWJTU Chinese word segmentation system. The accuracy achieves 95.80% when the method is tested in People's Daily January 1989 news data. Compared with other POS tagging methods, the method mentioned in the paper is better and accuracy is 88% in ambiguity words POS tagging experiment. This method also solves the problem of POS tag errors in the dictionary, insufficient training corpus in statistical machine learning method, ignoring context information of words. [4]

In this paper, extractive text summarization is obtained by using sentence ranking. The input file is first tokenized and stop words are removed to get the filtered text. The words that are preserved are considered as keywords and part of speech is assigned to each word(token).Weights are assigned to individual tokens and weighted frequency is calculated. The sum of the weighted frequencies and individual term ranks are calculated. The summarizer will extract the high weighted frequency sentences in order to find a summary of a document and the extracted summaries are converted into audio form. They have evaluated the performance of their system with human generated summary and came to a conclusion that their system provided a more effective summary. The proposed system provides better accuracy when compared to the traditional approach.[5]

III. PROBLEM DEFINITION

Our problem mainly deals with data having many sentences. People need to learn much from data. But they tend to want to spend less time while doing this. It is very difficult for human beings to manually extract the summary of a large document of text. There are plenty of text materials available on the Internet. So there is a problem of searching for relevant documents from the number of documents available, and absorbing relevant information from it. This is when our project comes into action. We aim to solve this problem by supplying them the summaries of the text from which they want to gain information. No matter how large is the data; our project will give a summary of it.

IV. PROPOSED SYSTEM METHODOLOGY

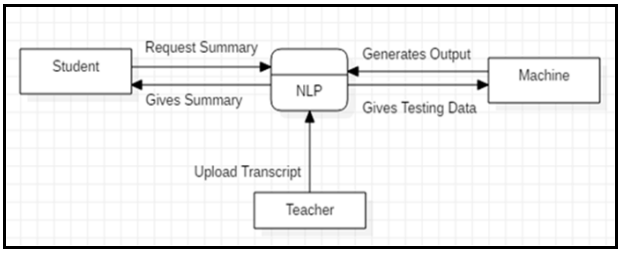

Fig. 1 Block diagram of one of the many applications of the system.

In the project, we've used the extractive approach to generate a summary. Our code first removes all stopwords, square brackets, extra spaces, special characters and digits as these are used frequently and cannot be taken as a base to get a summary. Once this is done, our code calculates the frequency (word count) of every word in the data. The word having the highest frequency is taken as a base in order to calculate the frequency (percentile) of every word. Once we calculate the frequency of every word, we calculate the score of every sentence. This is done by simply adding the frequency of every word in the sentence to get the score. We provide an option to the user to enter the number of sentences he wants in his summary. Based on the number of sentences the user wants, we provide the sentences having highest scores in the summary. The system works by assigning scores to sentences in the document to be summarized, and using the highest scoring sentences in the summary. Score values are based on features extracted from the sentence. A linear combination of feature scores is used. Almost all of the mappings from feature to score and the coefficient values in the linear combination are derived from a training corpus. We have ensured that our project can take any amount of data and provides a summary for the same. We have also designed our system in such a way that while processing the data, it removes stopwords, unnecessary words and provide a summary based on the relevant information in the data. Our system will be greatly beneficial to the end user as it will present the source text into a shorter version with semantics. The most important advantage of using a summary is that it provides us with maximum information in a short span of time. Our system will identify the most important meaningful information in a document or a set of related documents and compress them into a shorter version preserving its overall meanings.

V. SUMMARIZING ALGORITHM

The preferred spelling of the word "acknowledgment" in America is without an "e" after the "g". The algorithm we design is an amalgamation of three steps, these steps together produced the desired output. The steps are as follows:

1. Find the Highest Text Rank Sentences, using the Pagerank Algorithm with Cosine Similarity as the comparison parameter

```
def textrank():
    S = Create Similarity Matrix
    sentence_ranks = pagerank(S)
    Sort the sentence ranks in Descending Order
    Select Top 50% of the Sentences
    Summary_1 -> Join all the selected sentences.
```

2. We acquire a summary from the above step, now we use that summary and perform POS analysis. In POS analysis

for the entire text we find the proper nouns, calculate the number of occurrences of that noun and choose the top 50% Nouns and select the sentences that have the selected nouns present in them.

```
def POStag():
    words = Convert the text into words.
    POS = Perform POS Tagging on the words using NLTK
    df -> calculate the frequency of each noun
    #Remove stopwords from the list
    Table -> Create 'Noun' and 'Frequency' Table
    Arrange the Table in descending order of Frequency
    Select the Nouns with Top 25% Frequency in the Table
Summary_2 -> Join the sentences with selected nouns
```

3. Finally after we acquire the second summary from the algorithm using POS Tagging, we pass that summary for a final level of refinement. We convert the entire text to words, find the frequency of each word, now next we calculate the sentence score which is for a sentence we add the frequency corresponding to each word in the sentence. Finally the top 25% of the highest sentence scores are selected and joined to create a summary.

```
def sentencescore():
    Table -> 'Word' and 'Frequency' Table
    Sentence List = Divide Summary_2 into sentences
        for sentence in Sentence List:
            for word in Total Word List:
                if word in Table['Word']:
                    if First word of the Sentence:
                        Sentence Scores = frequency[word]
                    else:
                        Sentence Scores += frequency[word]
    Sort the Sentence Scores in Descending Order
    Select the Top 25% of the Sentence Score Table
    Summary_Final -> Join Selected Sentences.
```

4. The Algorithm in short is First Text Rank, the POS Tagging and Finally Sentence Score Algorithm.

For summarizing online websites we create the data using a beautiful soup package that extracts all the data in the <p> paragraph tags.

VI. PERFORMANCE EVALUATION PARAMETER

The algorithm output statistics of some sample texts are shown below in numerical format for better understanding of how the algorithm is generated and how brief the summary is.

1. Link 1
   https://en.wikipedia.org/wiki/Computer
   Original data – 1023 sentences
   Our summary – 20 sentences
2. Link 2
   https://en.wikipedia.org/wiki/Machine_learning
   Original data – 756 sentences
   Our summary – 19 sentences
3. Offline Summary
   Original data – 200 sentences
   Our summary – 13 sentences

Since it is a summary of the text, we cannot evaluate how precise it is. The method used is extractive and not abstractive there is no mathematical way to calculate preciseness of the summary. There are a few differences which we observed in the outputs we received for the same text using two different algorithms they are as follows:

1. Time taken for extractive summarisation is faster compared to the abstractive algorithm (i.e. RNN and LSTM) we implemented.
2. Abstractive Summarization shows promising results but the observed trend is that this method is data hungry and needs to be fed a lot of data to train, hence to increase the accuracy there has to be a compromise in computation time, whereas the extractive was faster and gave quite relevant output.
3. Deep Learning Algorithms such as RNN and LSTM give different output for different configurations of hyper parameters such as no epochs, learning rate, etc. hence every text has a unique value of hyper parameters to get optimum accuracy but it's not possible to tweak the parameters after the model is created. Hence the more adaptive solution found is extractive summarisation that shows best output according to the text given
4. RNN and LSTM follow semantic analysis where they follow a fixed set of grammar they are trained to , hence if no kind of data is present for the person whose data it is not trained the results may be variable. To overcome the problem the extractive analysis works on frequency as a metric hence it is document specific and also user specific where the authors favourite words and ideas are repeated and they are reflected in the summary

To make the analysis more prudent we conducted two tests to check which type of summarization is the best.

1. Machine / Mathematical Based Testing
2. Human based Testing

The results for the tests are shown below in a tabular form

Test 1: The first test involved taking 3 texts and having a human creating a summary for all three and then comparing them using cosine and jaccard similarity.

		Sample 1	Sample 2	Sample 3
Abstractive Summary	Cosine Similarity	0.95524	0.98992	0.96415
	Jaccard Similarity	0.96221	0.90426	0.93251
Extractive Summary	Cosine Similarity	0.96175	0.98963	0.97157
	Jaccard Similarity	0.97012	0.98847	0.97693

Table 1: Results of test 1

Test 2: The second test is generating 3 summaries and asking two humans who have read the text to judge the summaries using parameters such as Similarity, Briefness and Relevance.

		Sample 1	Sample 2	Sample 3
Relevance	Abstractive Summary	6	7	7
	Extractive Summary	8	6	9
Similarity	Abstractive	9	8	8

	Summary			
	Extractive Summary	8	7	8
Briefness	Abstractive Summary	7	6	6
	Extractive Summary	9	8	8

Table 2: Results of test 2

VII. REFERENCES

[1] N. S. Shirwandkar and S. Kulkarni, "Extractive Text Summarization Using Deep Learning," 2018 Fourth International Conference on Computing Communication Control and Automation (ICCUBEA), Pune, India, 2018, pp. 1-5.

[2] N. Moratanch and S. Chitrakala, "A survey on extractive text summarization," 2017 International Conference on Computer, Communication and Signal Processing (ICCCSP), Chennai, 2017, pp. 1-6. doi: 10.1109/ICCCSP.2017.7944061

[3] A. Jain, D. Bhatia and M. K. Thakur, "Extractive Text Summarization Using Word Vector Embedding," 2017 International Conference on Machine Learning and Data Science (MLDS), Noida, 2017, pp. 51-55. doi: 10.1109/MLDS.2017.12

[4] Z. Ye, Z. Jia, J. Huang and H. Yin, "Part-of-speech tagging based on dictionary and statistical machine learning," 2016 35th Chinese Control Conference (CCC), Chengdu, 2016, pp. 6993-6998. doi: 10.1109/ChiCC.2016.7554459

[5] J. N. Madhuri and R. Ganesh Kumar, "Extractive Text Summarization Using Sentence Ranking," 2019 International Conference on Data Science and Communication (IconDSC), Bangalore, India, 2019, pp. 1-3. doi: 10.1109/IconDSC.2019.8817040

[6] Inderjeet Mani and Eric Bloedorn, "Multi-document summarization by graph search and matching," AAAI/IAAI, vol. cmplg/ 9712004, pp. 622- 628, 1997.

[7] J. M. Conroy, and D. P. O'leary, "Text summarization via hidden markov models," In Proceedings of SIGIR '01, pp. 406-407, New York, NY, USA, 2001.

[8] Nenkova, A.(2011)."Automatic summarization, Foundations and Trends in Information Retrieval",5(2),103-233

[9] Gupta,V and Lehal,G.s (2010). "A survey of text summarization extractive techniques." Journal of Emerging Technologies in Web Intelligence,2(3),258-268

[10] J. Cheng, M. Lapata, "Neural summarization by extracting sentences and words," arXiv:1603.07252 [cs.CL]

[11] M. S. Patil, M. S. Bewoor, S. H. Patil, "A Hybrid Approach for Extractive Document Summarization Using Machine Learning and Clustering Technique," International Journal of Computer Science & Information Technology, Vol. 5, Issue 2, 2014, pp. 1584

[12] R. Nallapati, F. Zhai, B. Zhou, "SummaRuNNer: A Recurrent Neural Network based Sequence Model for Extractive Summarization of Documents," in Proc. AAAI Conference on Artificial Intelligence (AAAI), Computational and Language(cs.CL), arXiv:1611.04230v1 [cs.CL]

[13] F. Chen, K. Han, and G. Chen, "An approach to sentence-selection based text summarization," in TENCON'02. Proceedings. 2002 IEEE Region 10 Conference on Computers, Communications, Control and Power Engineering, vol. 1. IEEE, 2002, pp. 489-493.