iJRASET
International Journal For Research in Applied Science and Engineering Technology

INTERNATIONAL JOURNAL FOR RESEARCH

IN APPLIED SCIENCE & ENGINEERING TECHNOLOGY

Volume: 10 **Issue:** IV **Month of publication:** April 2022

DOI: https://doi.org/10.22214/ijraset.2022.40766

www.ijraset.com

Call: 08813907089 | E-mail ID: ijraset@gmail.com

International Journal for Research in Applied Science & Engineering Technology (IJRASET)
ISSN: 2321-9653; IC Value: 45.98; SJ Impact Factor: 7.538
Volume 10 Issue IV Apr 2022- Available at www.ijraset.com

Applying Text Rank to Build an Automatic Text Summarization Web Application

Rohit Parimoo[1], Rohit Sharma[2], Naleen Gaur[3], Nimish Jain[4], Sweeta Bansal[5]
[1, 2, 3, 4, 5]*Department of Computer Science Engineering, Inderprastha Engineering College, Ghaziabad, Uttar Pradesh*

Abstract: ***Automatic Text Summarization is one of the most trending research areas in the field of Natural Language Processing. The main aim of text summarization is to reduce the size of a text without losing any important information. Various techniques can be used for automatic summarization of text. In this paper we are going to focus on the automatic summarization of text using graph-based methods. In particular, we are going to discuss the implementation of a general-purpose web application which performs automatic summarization on the text entered using the Text Rank Algorithm. Summarization of text using graph-based approaches involves pre-processing and cleansing of text, tokenizing the sentences present in the text, representing the tokenized text in the form of numerical vectors, creating a similarity matrix which shows the semantic similarity between different sentences present in the text, representing the similarity matrix as a graph, scoring and ranking the sentences and extracting the summary.***

Keywords: Text Summarization, Unsupervised Learning, Text Rank, Page Rank, Web Application, Graph Based Summarization, Extractive Summarization

\

I. INTRODUCTION

Traditionally, summarization of text has been done manually by human beings and it has always been a time consuming and cumbersome task. In many applications where a reader might just need the important points of an article and not necessarily the most natural human written summary, the process of manually summarizing the text causes unnecessary wastage of time. Automatic text summarization can come in handy in these kinds of scenarios. The problem of automatic summarization of text can be solved using various approaches [1]. We are particularly going to focus on graph-based techniques of summarization of text. One such technique is the Text Rank algorithm [2] which works on the same principle as the PageRank algorithm [3] which is used to rank web pages in terms of their importance. Text Rank is a supervised learning general purpose NLP algorithm which can be used for the problem of automatic text summarization. It builds an undirected graph whose vertices represent the sentences in the text and the edge weights represent some sort of semantic or lexical similarity between those sentences like for instance, cosine similarity. It uses this graph to score sentences on the basis of their importance using the same approach as PageRank.

II. METHODOLOGY

Text Rank is an unsupervised key phrase extraction algorithm. It uses the structure of the text itself to identify key phrases that appear important to the text in the same way in which PageRank identifies important Web pages. First the text is split into sentences, following which the sentences are tokenized and some cleaning is performed on them like removing punctuations, stop words etc. Then these tokenized sentences are converted into vectors (word embeddings). On the bases of the vectors created a similarity matrix is drawn. The entry in the i^{th} row and the j^{th} column of this matrix represents the degree of similarity between sentence i and sentence j. This similarity value is some sort of lexical or semantic measure of similarity between two sentences like cosine similarity [4]. On the basis of the values of this similarity matrix an undirected graph is created, whose vertices represent the sentences in the text and its edge weights represent the degree of similarity between the sentences represented by the vertices connected by those edges. Then the PageRank algorithm is executed on this graph in order to score the sentences. After this, the sentences are ranked on the basis of their sentence score and a summary is created by concatenating the top n sentences. Where n is a whole number determined by the size of the summary required. A REST API is created to take the text to be summarized as an input and to output the summary generated using the summarization algorithm. A client-side user interface allows the users to enter the text as an input in a text area and passes that text to the backend REST API. It takes the response of the backend (i.e., the summarized text) and displays it to the user. It also allows the user to download the generated summary.

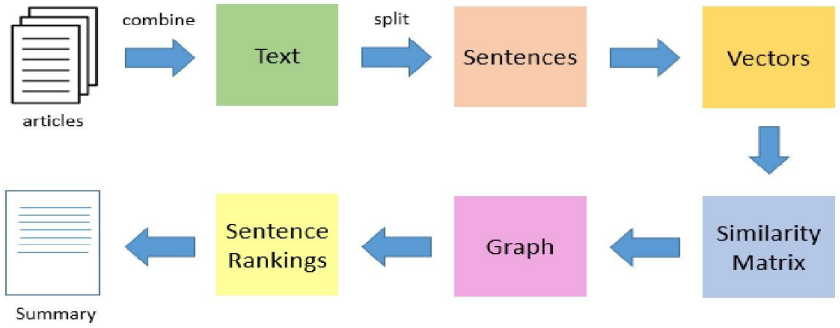


Fig. 1 Methodology of summarization

III. IMPLEMENTATION OF THE WEB APPLICATION

The Text Rank algorithm can be used to implement an end-to-end full stack automatic text summarization web application. Implementing such application will make it easier for any person with no technical expertise to automatically summarize any text like news articles, stories, novels or the content of any web page on the internet. Both the frontend and the backend of the application need to be implemented.

A. Server-Side Code

The server-side part of the web application should be implemented as a Restful API that can take some text and the length the of the required summary as an argument and fetches the summary in JSON format. Under the hood, the server-side code will contain the entire implementation of the text rank algorithm including all the pre-processing and the data cleaning steps. It is important that the algorithm and the entire server-side code should be optimized as much as possible using appropriate data structures and other time and space optimization techniques.

B. Database

To create a complete web application the users should be provided with the option to create an account and save their summaries. Here is when a database becomes useful. Any standard relational database management system can be used to store the identity of users and the summaries generated by them. This database will also be used to store the login IDs and the encrypted passwords of the users.

C. Client-Side Code

The client-side code will be a simple user interface that will allow the user to enter the text to be summarized in a text area provided and also to choose the length of the required summary. Then it will make API calls to the backend REST API in order to fetch the required summary in JSON format. It would then render that summary to the user and also provide options to download that summary in various common text formats. The client-side code will also provide Sign Up and Login pages to the users. A functionality to display and download the saved summaries of users will also be included in the UI.

D. Pre-Processing

With regard to this application, pre-processing involves splitting the text into sentences, tokenizing the sentences, removing any unnecessary elements like stop words and punctuations in order to reduce any noise. Stop words are commonly used words such as a, an, the etc. that do not help in determining the semantic or lexical similarity value between two sentences [5]. So, such unnecessary words must be removed in order to reduce noise during summarization.

E. Optimising The Sentence Ranking Step

The sentence ranking step can be optimized further by using a max heap data structure to which the sentences and their respective scores can be added. A max heap will have O(1) time complexity for fetching the sentence with the highest score, O(log n) for removing that sentence and O(log n) for adding a new sentence and its score. So, it can be a pretty efficient data structure for sentence ranking.

F. Security

Security is one of the areas of prime concern in the context of web applications. Certain steps can be taken to enhance the security of the text summarization web application. Cross-site request forgery can be prevented using secret CSRF tokens generated by the server-side code while any request is made by the client-side code [6]. Also, SQL injection attacks can be prevented by using the server-side frameworks ORM's for accessing the database as they are specifically designed to prevent such attacks. Password must be encrypted at the client side only using a strong encryption algorithm like for instance the AES algorithm.

IV. EVALUATION

The quality of the summaries created by any text summarization system can be evaluated through a group of metrics known as ROUGE [7]. ROUGE or Recall-Oriented Understudy for Gisting Evaluation is a group of metrics used for evaluating text summarization techniques. These metrics compare the computer-generated summary of a text with a human made reference summary of that text. ROUGE metrics mainly use the number of matching n-grams in the computer-generated summary and the human made reference summary with the exception of ROUGE-L metric which uses the number of characters in the longest common subsequence in the two summaries in order to compare them. ROUGE metrics can be classified into three main types as depicted by the table below.

TABLE I. ROUGE METRICS

	ROUGE-1	ROUGE-2	ROUGE-L
RECALL	$\frac{\text{No. of Matching Unigrams}}{\text{Size of Reference Summary}}$	$\frac{\text{No. of Matching bi-grams}}{\text{Size of Reference Summary}}$	$\frac{\text{Length of Longest Common Subsequence}}{\text{Size of Reference Summary}}$
PRECISION	$\frac{\text{No.of Matching Unigrams}}{\text{Size of Generated Summary}}$	$\frac{\text{No.of Matching bi-grams}}{\text{Size of Generated Summary}}$	$\frac{\text{Length of Longest Common Subsequence}}{\text{Size of Generated Summary}}$

V. RESULT

Using the Text Rank algorithm with all the pre-processing steps discussed above we can build an end-to-end web application that can enable users to summarize any kind of text like news articles, web page content, novels, stories etc. It acts as a general-purpose automatic text summarization system. It also allows users to set the approximate length of the required summary and generates a summary close to that text. Also, the optimized code of the Text Rank algorithm can summarize a large text like a novel in at most a few seconds. The fast execution time of this algorithm can be useful in making this web application highly scalable. Automatic text summarization is one of the common areas of research in the field of Natural Language Processing. Many approaches can be used to solve it. Using this paper, we have discussed the process of building a Text Summarization Web Application using the Text Rank algorithm.

VI. CONCLUSION

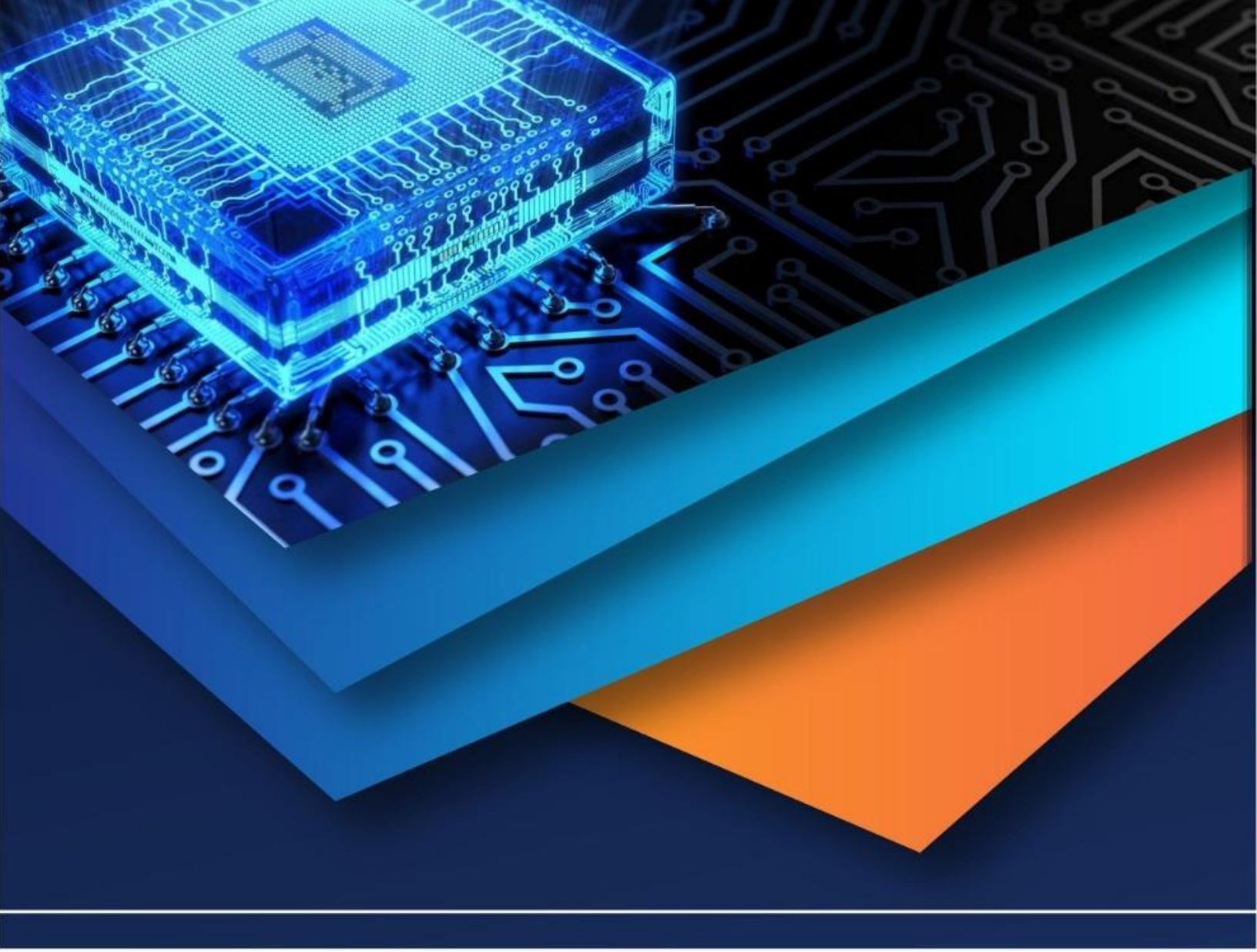
We have seen, how the Text Rank algorithm which is a graph based unsupervised learning algorithm as discussed above can be used to develop and end-to-end high performance web application that performs automatic summarization on the text entered by the user. It results in a general-purpose text summarization web application that can be used to summarize any kind of text. It lets the user decide the size of the required summary. So, we have managed to completely automate the time consuming and tedious task of text summarization.

VII. ACKNOWLEDGMENT

We would like to thank our project supervisor and mentor Dr. Sweeta Bansal for supporting and guiding us through all the stages of writing this research paper. Her guidance and support made this work possible.

REFERENCES

[1] Mehdi Allahyari et al. (2017). Text Summarization Techniques: A Brief Survey. International Journal of Advanced Computer Science and Applications
[2] Mihalcea, Rada & Rada, & Tarau, Paul & Paul. (2004). TextRank: Bringing Order into Texts.
[3] Brin and L. Page. 1998. The anatomy of a large-scale hypertextual Web search engine. Computer Networks and ISDN Systems.
[4] Gunawan, Dani & Sembiring, C & Budiman, Mohammad. (2018). The Implementation of Cosine Similarity to Calculate Text Relevance between Two Documents. Journal of Physics: Conference Series. 978. 012120. 10.1088/1742-6596/978/1/012120.
[5] Sarica, Serhad & Luo, Jianxi. (2020). Stopwords in Technical Language Processing.
[6] Chen, Boyan & Zavarsky, Pavol & Ruhl, Ron & Lindskog, Dale. (2011). A Study of the Effectiveness of CSRF Guard. 1269-1272. 10.1109/PASSAT/SocialCom.2011.58.
[7] Lin, Chin-Yew. (2004). ROUGE: A Package for Automatic Evaluation of summaries. Proceedings of the ACL Workshop: Text Summarization Braches Out 2004. 10.



10.22214/IJRASET

45.98

IMPACT FACTOR: 7.129

IMPACT FACTOR: 7.429

INTERNATIONAL JOURNAL FOR RESEARCH

IN APPLIED SCIENCE & ENGINEERING TECHNOLOGY