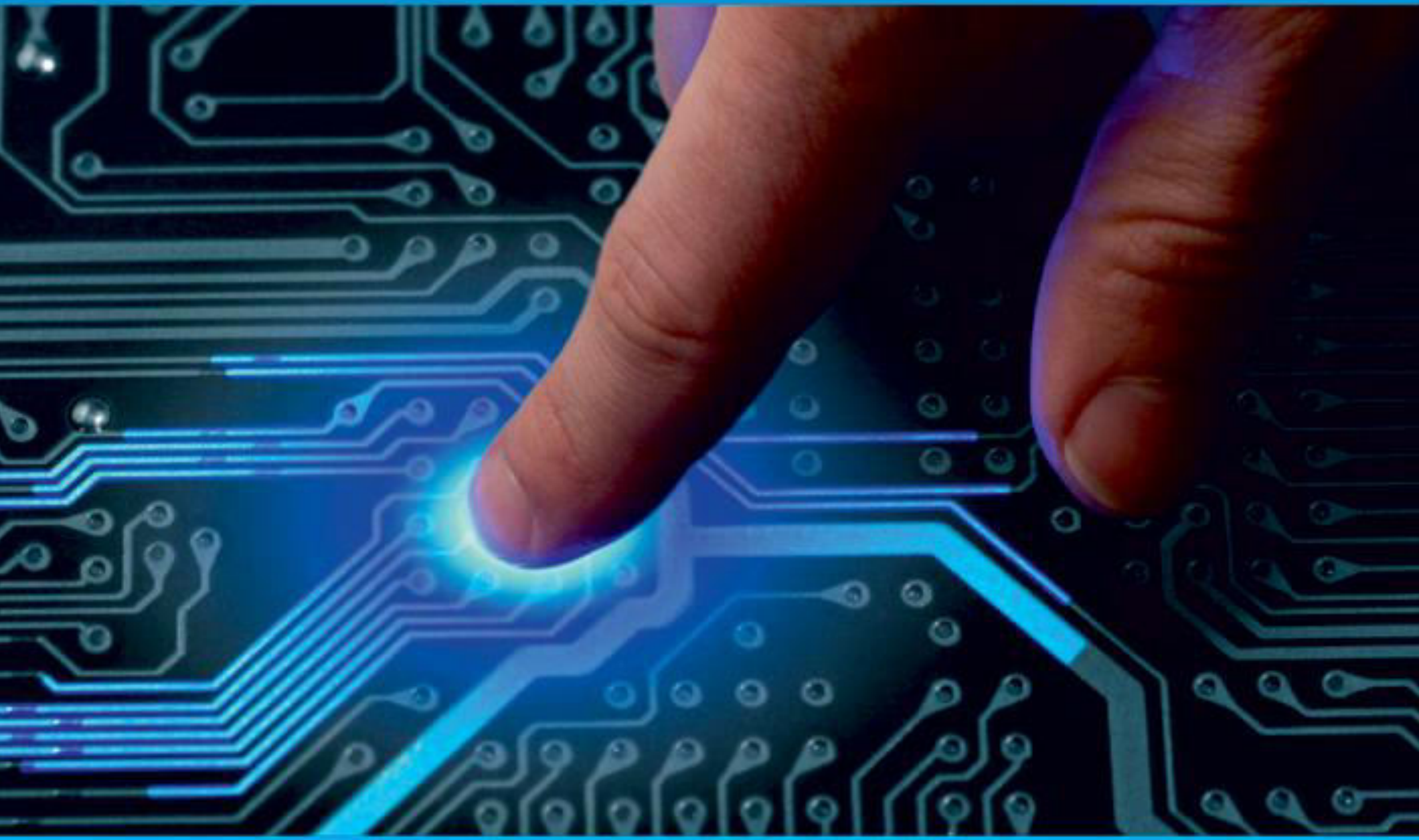
IJIRCCE

e-ISSN: 2320-9801 | p-ISSN: 2320-9798



INTERNATIONAL JOURNAL OF INNOVATIVE RESEARCH

IN COMPUTER & COMMUNICATION ENGINEERING

Volume 11, Issue 3, March 2023

INTERNATIONAL
STANDARD
SERIAL
NUMBER
INDIA

Impact Factor: 8.379

International Journal of Innovative Research in Computer and Communication Engineering

| e-ISSN: 2320-9801, p-ISSN: 2320-9798| www.ijircce.com | |Impact Factor: 8.379 |

|| Volume 11, Issue 3, March 2023 ||

| DOI: 10.15680/IJIRCCE.2023.1103088 |

Text Summarizer Using NLP

Sk. Mulla Almas, Praneeth Oggu, Yenibara Ajay, Syed Mohammmed Iqbal

Assistant Professor, Department of IT, Vasireddy Venkatadri Institute of Technology, Guntur, Andhra Pradesh, India

UG Student, Department of IT, Vasireddy Venkatadri Institute of Technology, Guntur, Andhra Pradesh, India

UG Student, Department of IT, Vasireddy Venkatadri Institute of Technology, Guntur, Andhra Pradesh, India

UG Student, Department of IT, Vasireddy Venkatadri Institute of Technology, Guntur, Andhra Pradesh, India

ABSTRACT: In the modern Internet age, textual data is ever increasing. Need some way to condense this data while preserving the information and meaning. We need to summarize textual data for that. Text summarization is the process of automatically generating natural language summaries from an input document while retaining the important points. It would help in easy and fast retrieval of information. Text summarization condenses a large amount of text into a shorter version while retaining the main points and ideas, saving time and improving comprehension. Additionally, text summarization can help to reduce bias and will become an essential tool for staying informed and making informed decisions as the amount of information available continues to grow. Text summarization using natural language processing (NLP) is a technique that uses algorithms to automatically generate a summary of a text. The proposed system uses NLP model techniques to summarize the information that allows the users to condense a larger piece of text and articles into a summarized form preserving the meaning of the text. It is implemented and deployed as Web API using flask deployment. It can be used to compare different summarised texts in order to determine the accuracy and expected time required to summarise the content.

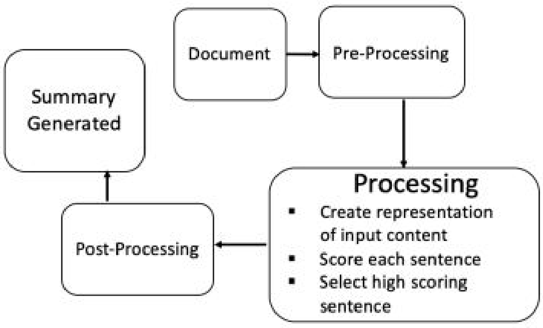
KEYWORDS: Sentence extraction, Text summarization, NLP, Keyword extraction

I. INTRODUCTION

Text summarization using natural language processing (NLP) is a technique that uses algorithms to automatically generate a summary of a text. NLP-based summarization can be done using extraction or abstraction methods. Extraction methods involve selecting the most important sentences or phrases from the original text, while abstraction methods involve generating new sentences that capture the main ideas of the original text. NLP-based summarization can be particularly useful for handling large amounts of information and can help to save time and improve comprehension.

To create summaries, extractive summarization systems copy significant portions of the original text and then mix those portions and sentences. Sentence importance is determined by linguistic and statistical characteristics.

Text summarization is becoming increasingly important in many industries due to the overwhelming amount of information available.NLP is a group of techniques that can be used to analyse text and extract its important elements. These methods can be applied to a text to extract keywords, themes, and main ideas. NLP may be used to quickly and accurately summarise the text and extract all of the key information.



II. RELATED WORK

[1] This paper “Extractive text summarization using neural networks”, They presented an entirely data-driven method for automatically summing up text. Without having access to any linguistic data, they created and tested the model on standard datasets, which produced results comparable to those of state-of-the-art models. Finding some drawbacks in their paper they assumed that the length of the page should be less than the length of the page.

[2] This paper “Get to the point: Summarization with pointer-generator networks”, By producing summaries based on the source text, they first pre-train the generative model. They then use the positive examples from the manually created summaries and the negative examples from the pre-trained generator to pre-train the discriminator. Generator and discriminator are alternately trained after pre-training.

[3] This paper “Generative adversarial network for abstractive text summarization”, They use the CNN/Daily Mail dataset, which consists of news stories (39 sentences on average) paired with multi-sentence summaries, to test their model, and they demonstrate that it performs at least 2 ROUGE points better than the state-of-the-art abstractive system.

[4] This paper “"Latent dirichlet allocation”, Each piece of information in the corpus is reduced to a vector of real values that each reflect ratios of counts. After adequate normalisation, this term frequency count is compared to an inverse document frequency count, which estimates the number of occurrences of a word in the entire corpus. A term-by-document matrix X with columns containing the tf-idf values for each document in the corpus is the final output. The tf-idf system reduces texts of any length to lists of numbers with set lengths. for each document in the corpus is the final output. The tf-idf system reduces texts of any length to lists of numbers with set lengths.

III. PROPOSED ALGORITHM

Text summarization is a technique used to create a shorter and more concise version of a larger text document. NLP (Natural Language Processing) is a subfield of artificial intelligence that deals with the interaction between computers and humans using natural language.The main steps involved in building a text using NLP are as follows:

1. Text Preprocessing

The first step is to preprocess the text data by removing unwanted charactecters, numbers and special symbols. The text is then tokenized into words and sentences using NLP techniques.

2. Sentence Scoring

The next step is to assign a score to each sentence in the text based on its relevance to the overall meaning of the document. This can be done using various techniques such as TF-IDF (Term Frequency-Inverse Document Frequency), TextRank, or LSA (Latent Semantic Analysis).

3. Sentence Selection

Once the sentences have been scored, the next step is to select the most important sentences that best represent the overall meaning of the document. This can be done by selecting the top-ranked sentences based on their scores or by setting a threshold score and selecting all sentences above that threshold.

4. Summary Generation

Finally, the selected sentences are combined to generate a summary of the text document. This summary should be a shorter version of the original text, containing only the most important information.

In conclusion, building a text summarizer using NLP involves text preprocessing, sentence scoring, sentence selection, and summary generation. By using NLP techniques, it is possible to create a more accurate and concise summary of a larger text document.

Tokenization

Tokenization is the process of dividing text into a list of tokens from a string of text. Tokens can be thought of as components, such as a word in a sentence or a sentence in a paragraph. To make use of this, we developed a function that accepts an article's text, tokenizes each phrase (dataframe rows), builds a vocabulary free of stop words for the specific content (dataframe columns), and then assigns TF-IDF weights to each word in the vocabulary for each sentence.

Textrank

Textrank is an algorithm inspired by Google’s PageRank algorithm that helps identify key sentences from a passage (Mihalcea, Rada, and Paul Tarau, 2004). The idea behind this 4 algorithm is that the sentence that is similar to most other sentences in the passage is probably the most important sentence in the passage. Using this idea, one can create a graph of sentences connected with all the similar sentences and run Google’s PageRank algorithm on it to find the most important sentences. These sentences would then be used to create the summary.

Spacy

Spacy also includes a built-in feature for text summarization, which uses a machine learning approach to generate summaries. The summarization algorithm in Spacy is based on the TextRank algorithm, which is an unsupervised algorithm for extractive summarization. It uses a graph-based approach to score sentences based on their importance and relevance to the main topic. Spacy is a powerful and efficient library for text summarization, allowing you to generate accurate and relevant summaries of your input text using machine learning techniques.

Since the summarization follows extractive text summarization it uses textrank algorithm which is an unsupervised algorithm. The sentences in a document are viewed by the TextRank algorithm as nodes in a graph. Similar to how PageRank determines key web sites based on links to other pages, it then identifies the important sentences in the document based on their relationships with other phrases. The TextRank algorithm's fundamental steps are as follows:

Step 1:The input text should be tokenized into sentences and words.

Step 2:Use a similarity metric, like cosine similarity, to determine how similar each pair of words are.

Step 3:Transform the sentence similarity matrix into a graph in which each sentence is represented by a node and the edges represent similarity between sentences.

Step 4: To order the sentences in the network according to importance, apply the PageRank algorithm.

Step 5: Choose the best sentences as input for text summary

TF-IDF (Term Frequency — Inverse Document Frequency)

Term Frequency Inverse Document Frequency of records is referred to as TF-IDF. It can be summed up as determining how pertinent a word is to a corpus or series of words in a text. The frequency of a term in the corpus offsets the meaning increase that occurs when a word appears more frequently in the text (data-set). Using cosine similarity, TF-IDF is applied.

Each terms are given weights using the TF-IDF (Term Frequency — Inverse Document Frequency) algorithm based on how distinctive they are in comparison to the document's total lexicon. Higher weight (more distinctive) words frequently have greater significance or give the paper a deeper meaning.

IV. PSEUDO CODE

```
Define function Stext_summarizer(raw_docx):
Step 1:  Create a Spacy object from the raw text input
Step 2:  Create a list of stop words
Step 3: Create an empty dictionary for word frequency
Step 4: For each word in the Spacy object,
    i.    If the word is not a stop word and not in the dictionary, add it with a frequency of 1
    ii.   If the word is already in the dictionary, increase its frequency by 1
Step 5: Calculate the maximum frequency from the dictionary
Step 6: For each word in the dictionary, divide its frequency by the maximum frequency to normalize the values
Step 7: Create a list of sentences from the Spacy object
Step 8: Create an empty dictionary for sentence scores
    i.    For each sentence in the list of sentences,
        i.    For each word in the sentence,
            a. If the word is in the word frequency dictionary and its sentence length is less than 30,
        i.   If the sentence is not in the dictionary, add it with the frequency of the word
        ii.  If the sentence is already in the dictionary, increase its frequency by the frequency of the word
Step 9: Select the top 7 sentences based on their scores using heapq nlargest function
Step 10: Convert the selected sentences to a list of strings
  Step 11: Join the list of strings to create the final summary
  Step 12:  Print the original document, the length of the document, the summarized document, and the length of the summary
```

V. Results

Fig 1. The summary homepage allows users to paste their material for summarising by either pasting the needed text or copying and pasting the webpage's source url. Fig 2.This is one of the methods that users can enter their text by copying and pasting the text that has to be summarised into the text box, which then displays the summarised text along with the reading times of the original text and the summarised text. Fig 3. By copying the website's url and pasting it into the text box on the input page, the summary input can be entered. Fig 5. Several libraries that display diverse summarised texts can be used to compare the summarised result. The libraries are spacy summarizer, genism summarizer, nltk and sumy lexrank.

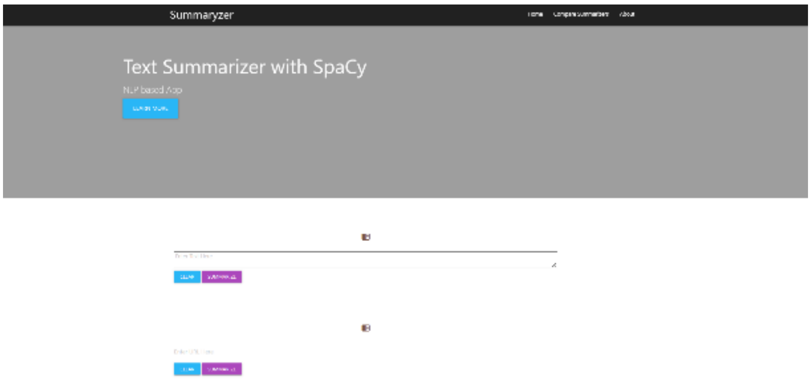
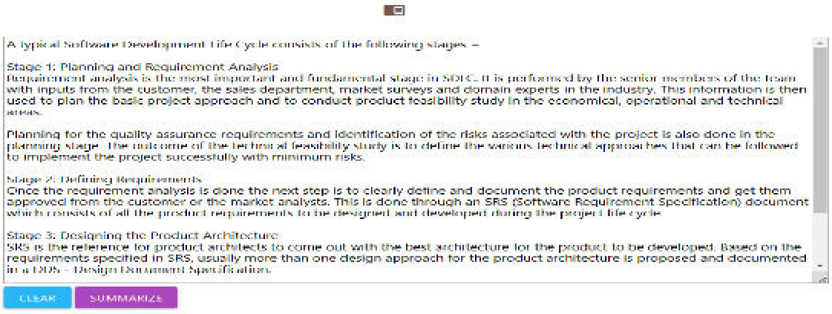


Fig 1 . Home page GUI



A typical Software Development Life Cycle consists of the following stages −

Stage 1: Planning and Requirement Analysis
Requirement analysis is the most important and fundamental stage in SDLC. It is performed by the senior members of the team with inputs from the customer, the sales department, market surveys and domain experts in the industry. This information is then used to plan the basic project approach and to conduct product feasibility study in the economical, operational and technical areas.

Planning for the quality assurance requirements and identification of the risks associated with the project is also done in the planning stage. The outcome of the technical feasibility study is to define the various technical approaches that can be followed to implement the project successfully with minimum risks.

Stage 2: Defining Requirements
Once the requirement analysis is done the next step is to clearly define and document the product requirements and get them approved from the customer or the market analysts. This is done through an SRS (Software Requirement Specification) document which consists of all the product requirements to be designed and developed during the project life cycle.

Stage 3: Designing the Product Architecture
SRS is the reference for product architects to come out with the best architecture for the product to be developed. Based on the requirements specified in SRS, usually more than one design approach for the product architecture is proposed and documented in a DDS - Design Document Specification.

Fig 2. Input as Text

| e-ISSN: 2320-9801, p-ISSN: 2320-9798| www.ijircce.com | |Impact Factor: 8.379 |

|| Volume 11, Issue 3, March 2023 ||

| DOI: 10.15680/IJIRCCE.2023.1103088 |

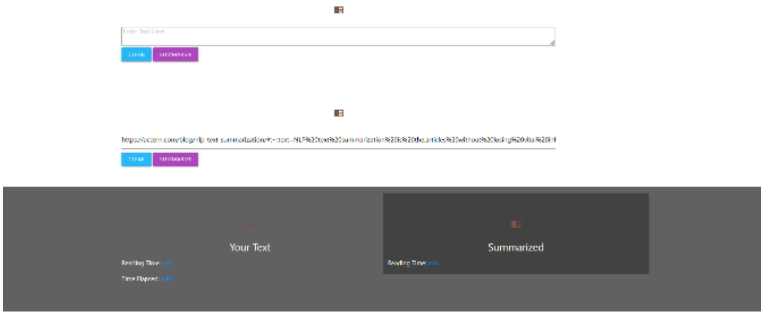


Fig 3. Input as Link

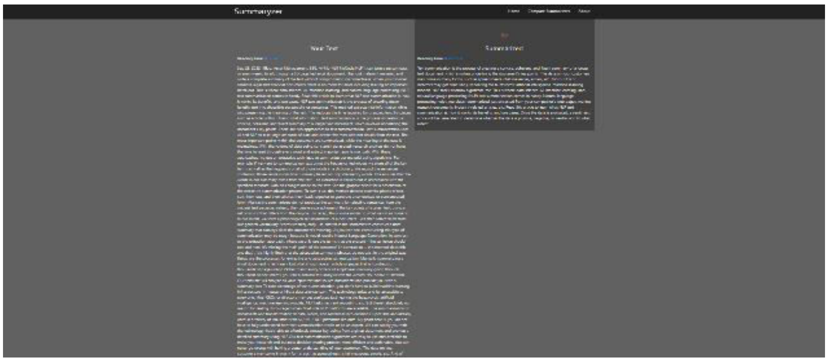
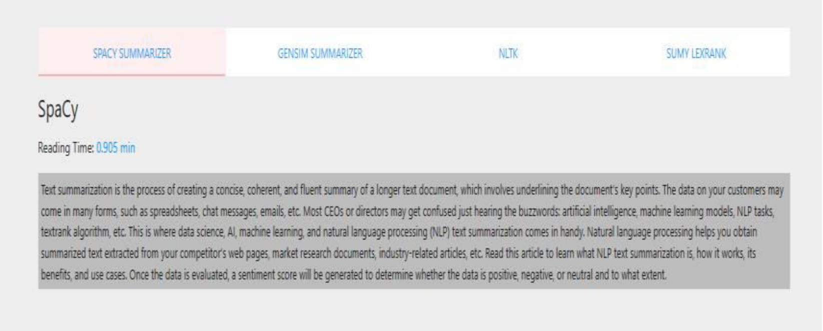


Fig 4. Summarized Output



SpaCy

Reading Time: 0.905 min

Text summarization is the process of creating a concise, coherent, and fluent summary of a longer text document, which involves underlining the document's key points. The data on your customers may come in many forms, such as spreadsheets, chat messages, emails, etc. Most CEOs or directors may get confused just hearing the buzzwords: artificial intelligence, machine learning models, NLP tasks, textrank algorithm, etc. This is where data science, AI, machine learning, and natural language processing (NLP) text summarization comes in handy. Natural language processing helps you obtain summarized text extracted from your competitor's web pages, market research documents, industry-related articles, etc. Read this article to learn what NLP text summarization is, how it works, its benefits, and use cases. Once the data is evaluated, a sentiment score will be generated to determine whether the data is positive, negative, or neutral and to what extent.

Fig 5. Summarizers Comaprison page

VI. Conclusion and Future Work

In conclusion, the enormous amount of information available online has led to an increase in the use of text summarization tools that apply natural language processing (NLP) methods. The goal of NLP-based text summarizers is to provide a condensed version of a lengthy text that nevertheless conveys the main ideas and content. The proposed system can be used that can better understand the context of the text and identify the most important information. In addititiion to this, the The proposed approach employs abstractive summarization, which selects the key passages from the source text and rephrases them so that they appear in the summarised text.

References

1. Sinha, Aakash, Abhishek Yadav, and Akshay Gahlot. "Extractive text summarization using neural networks." arXiv preprint arXiv:1802.10137 (2018).
2. Peter J. Liu, and Christopher D. Manning. "Get to the point: Summarization with pointer-generator networks." arXiv preprint arXiv:1704.04368 (2017).

3. Liu, Linqing, et al. "Generative adversarial network for abstractive text summarization." Proceedings of the AAAI Conference on Artificial Intelligence. Vol. 32. No. 1. 2018.
4. Blei, David M., Andrew Y. Ng, and Michael I. Jordan. "Latent dirichlet allocation." Journal of machine Learning research 3.Jan (2003): 993-1022.

INTERNATIONAL JOURNAL OF INNOVATIVE RESEARCH

IN COMPUTER & COMMUNICATION ENGINEERING