BRIEFIFY - A Text Summarization tool

## MINI PROJECT REPORT

***Submitted by***

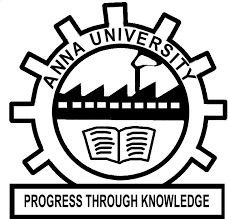
|  |  |
| --- | --- |
| **Sivanantham D** | **210701250** |
| **Gopal K** | **210701517** |
|  |  |

***in partial fulfillment for the award of the degree of***

**BACHELOR OF ENGINEERING**

***in***

**COMPUTER SCIENCE AND ENGINEERING**



**RAJALAKSHMI ENGINEERING COLLEGE, CHENNAI ANNA UNIVERSITY:: CHENNAI 600 025**

## APRIL 2024

**RAJALAKSHMI ENGINEERING COLLEGE, CHENNAI**

**BONAFIDE CERTIFICATE**

Certified that this Report titled “**BREFIFY – A text summarization tool**” is the bonafide work of **“Sivanantham D (210701250), Gopal K (210701517)** who carried out the work under my supervision. Certified further that to the best of my knowledge the work reported herein does not form part of any other thesis or dissertation on the basis of which a degree or award was conferred on an earlier occasion on this or any other candidate.

|  |
| --- |
| **SIGNATURE** |
| Dr. M. RAKESHKUMAR , M.E.,Ph.D., |
| **Assistant Professor,** |
| Department of Computer Science and Engineering, |
| Rajalakshmi Engineering College, |
| Chennai – 602015 |

Submitted to Mini Project Viva-Voce Examination held on

**Internal Examiner External Examiner**

## ABSTRACT

In order to develop an efficient solution in the summarizing of long documents, the Text summarization system is introduced. This is done using cosine similarity in combination with the PageRank algorithm to determine the semantic closeness and relevance of each of the key sentences. The utilisation of the Streamlit web application entails the ability of the users to upload the text files in connection with the utility thereby providing the users with an output commonly in the form of a summarised version which the ordinary users would enable them to understand and make the right decisions in the shortest time possible. As proved from several experiments on number of different data sets, it also generates as accurate, summarised and even coherent. For this reason, there are numerous applications of Information condensation with this particular strategy which include but not limited to content curation and document summarization. On its background of cutting-edge computations and accessible interface, which offers all essential constituent components, the provided system fully meets the need for the modern work and studies, in the sphere of academic and practical performance features. The steps in the data collection, data preprocessing, distance calculations, graph construction, ranking of the mentioned sentential segments, and the summary creation form the BRIEFIFY approach details. Each stage is designed to ensure that the summaries produced have both clarity of perceiving and semantic value. The utility and efficiency of the system have been further affirmed by its stochastic optimization on a plethora of datasets.

## ACKNOWLEDGEMENT

Initially we thank the Almighty for being with us through every walk of our life and showering his blessings through the endeavour to put forth this report. Our sincere thanks to our Chairman **Mr. S.MEGANATHAN, B.E, F.I.E.**, our Vice Chairman **Mr. ABHAY SHANKAR MEGANATHAN, B.E., M.S.,** and our

respected Chairperson **Dr. (Mrs.) THANGAM MEGANATHAN**, **Ph.D.,** for providing us with the requisite infrastructure and sincere endeavoring in educating us in their premier institution.

Our sincere thanks to **Dr. S.N. MURUGESAN, M.E., Ph.D.,** our beloved Principal for his kind support and facilities provided to complete our work in time. We express our sincere thanks to **Dr. P. KUMAR, Ph.D.**, Professor and Head of the Department of Computer Science and Engineering for his guidance and encouragement throughout the project work. We convey our sincere and deepest gratitude to our internal guide, **Dr. RAKESH KUMAR M, M.E., Ph.D.,** Assistant Professor Department of Computer Science and Engineering for his encouragement and guiding us throughout the project to build our project.

**Sivanantham D - 210701250**

**Gopal K -210701517**

**TABLE OF CONTENTS**

|  |  |  |
| --- | --- | --- |
| **CHAPTER NO.** | **TITLE** | **PAGE NO.** |
|  | **ABSTRACT** | **iii** |
|  | **ACKNOWLEDGEMENT** | **iv** |
| **1.** | **INTRODUCTION** | **7** |
|  | 1.1 GENERAL | 7 |
|  | 1.2 OBJECTIVE | 7 |
|  | 1.3 EXISTING SYSTEM | 7 |
|  | 1.4 PROPOSED SYSTEM | 8 |
| **2.** | **LITERATURE SURVEY** | **8** |
| **3.** | **SYSTEM DESIGN** | **10** |
|  | 3.1 DEVELOPMENT ENVIRONMENT | 10 |
|  | 3.1.1 HARDWARE SPECIFICATIONS | 10 |
|  | 3.1.2 SOFTWARE SPECIFICATIONS | 10 |
|  | 3.2 SYSTEM DESIGN | 11 |
|  | 3.2.1 ARCHITECTURE DIAGRAM | 11 |

|  |  |  |
| --- | --- | --- |
| **4.** | **PROJECT DESCRIPTION** | **12** |
|  | 4.1 MODULES DESCRIPTION | 12 |
| **5.** | **IMPLEMENTATION AND RESULTS** | 13 |
|  | 5.1 IMPLEMENTATION | 13 |
|  | 5.2 OUTPUT SCREENSHOTS | 14 |
| **6.** | **CONCLUSION AND FUTURE ENHANCEMENT** | **22** |
|  | 6.1 CONCLUSION | 23 |
|  | 6.2 FUTURE ENHANCEMENT | 23 |
|  | **REFERENCES 24** | |

## CHAPTER 1 INTRODUCTION

### GENERAL

Since the amount of digital text data has increased exponentially in an era where textual data has increased, knowledge management and information retrieval have faced difficult times. This is where defining the main idea is important, or breaking up longer pieces into shorter ones, as this prevents readers from becoming overwhelmed with information. In addition to saving a ton of time, this enhances production in terms of efficient comprehension and decision-making. By evaluating the supplied data, fields such as natural language processing have witnessed or produced improved summaries that can quickly and precisely capture the content.

### OBJECTIVE

The main aim of this project is a unique text summary tool that uses natural language processing (NLP) techniques to identify and extract important information from long documents and create a synopsis of it. The program aims to facilitate readers' speedy and effortless familiarization with vast amounts of textual data by utilizing technologies such as cosine similarity and PageRank for semantic analysis. Moreover, the concept behind putting the text processing algorithms into practice and making the project useful is to provide an intuitive user interface that will accept text for summarization and deliver the output in an easily readable format.

### EXISTING SYSTEM

While there are two subsets of text summarization methods which include abstraction-based and extraction-based solutions. SUMMRY, QuillBot, and Grammarly are some of the prolific tools available in the market. The summaries are produced with the help of several characteristic features of mixed NLP tools, and they differ from each other by facilities. However, most of the previously developed systems work has some drawbacks regarding the characteristics of summaries quality and coherence. Moreover, the implementation of these Systems into easily-navigable structures that allow for upload of documents and quick accessibility of summaries is inadequate.

### PROPOSED SYSTEM

BRIEFIFY, improves upon the functionality of earlier text summarization systems by introducing techniques not used in those programs. Sentence splitting, tokenization, and stop word removal are some of the techniques the system uses to process data that comes from various text sources. The text search system creates a similarity matrix by calculating the cosine similarities between the disliked and the other sentences. The PageRank algorithm then ranks the common sentences according to the overall value's relevance.

## CHAPTER 2 LITERATURE SURVEY

This paper [1] As the amount of textual content available online becomes more and more of a worry, text summarizing is becoming acknowledged as a crucial tool for managing the information overload. This essay will examine the difficulties associated with manual summary and the requirement for automatic summarization creation by systems capable of condensing a text while maintaining the article's meaning. It looks at the links between text mining techniques and summarizing methodologies, classes them, and evaluates performance using standards deemed necessary.

To address the complexity of summarizing Bangla text, demonstrating its effectiveness through experimental results. this paper [2] Text summary becomes an indispensable tool in the age of expanding data, helping to gain succinct information, cut down on reading time, and streamline research work. However, because of their complexity, summarization is difficult in languages like Bangla. In order to close this gap, this research suggests an extraction- based summarizing strategy designed specifically for Bangla language. Its efficacy is demonstrated by experimental results, which point to potential future developments in smart machine technology for industry 4.0.

This paper [3] describes the proposes a three-stage automatic rule reduction technique-based text categorization and summarizing approach: token creation, feature identification, and categorization/summarization. When evaluated with sample input texts, the built text analyzer yields impressive results. The effectiveness of the method and parameter choices for text classification have been confirmed by extensive testing. Many real-world uses, such as word sense disambiguation, information extraction, web resource categorization, and document retrieval, could benefit from this study.

This paper discusses a proposed text summarization technique in NLP that prioritizes phrase selection, utilizes structured diagrams for text representation [4] In the field of Natural Language Processing (NLP), this study suggests a text summary technique that emphasizes phrase selection from the source material. It uses structured diagrams to represent unstructured texts and preprocesses them so that different feature extraction techniques can be used. Without requiring in-depth language expertise, the method is flexible enough to work with a variety of languages and evaluates sentence relevance by linear weighting.

This review paper [5] Selecting important information is made more difficult by the exponential rise of data. This is addressed by text summarizing using Natural Language Processing (NLP), which reduces data without losing its essential meaning. This paper suggests a model that uses NLTK for sentiment analysis on news content with the goal of effectively extracting and presenting pertinent information to users.

This study [6] In an effort to save users time and effort, the quickly expanding area of automated text summarization (ATS) seeks to automatically condense enormous text volumes. This overview examines the issues and developments in ATS and covers extractive, abstractive, and hybrid strategies developed since the 1950s. Even with these advances, machine summaries still often differ from human ones. The paper offers a thorough examination of ATS, covering methods, challenges, applications, and evaluation metrics, for scholars studying the topic.

Aimed at aiding the Police Department with proper crime forecasting, this study [7] Automatic text summarization is the practice of employing either extractive or abstractive approaches to extract pertinent information from a text. This work presents a novel statistical method for extractive summarization of single documents. High-ranking sentences are selected from a list of sentences ranked using preset weights to provide a high-quality summary that can be recorded as audio.

This review paper [8] Text summarizing is the process of extracting important information from a text and condensing it into a brief summary in order to satisfy the increasing demand for simplicity in the news, business, and research domains. By contrasting extractive and abstractive summarization algorithms, this work aims to evaluate the implementation time, accuracy, and human-like quality of generated summaries. The study uses manual inspection and summary assessment to evaluate the benefits and drawbacks of each approach.

This paper [9] The exponential growth of textual data across multiple languages in the Big Data era makes effective text summarizing approaches essential. Large texts are automatically condensed into manageable chunks by the use of extracting complete sentences (Extractive) or reformulating them (Abstractive). There is a dearth of methodologies for Indian languages compared to those for English and other European languages. This article suggests a machine learning-based method for summarizing Hindi texts and addresses current research issues. Additionally, it examines text summarization methods in both Indian and foreign languages.

## CHAPTER 3 SYSTEM DESIGN

### DEVELOPMENT ENVIRONMENT

* + 1. **HARDWARE SPECIFICATIONS**

This project uses minimal hardware but in order to run the project efficiently without any lack of user experience, the following specifications are recommended

**Table 3.1.1** Hardware Specifications

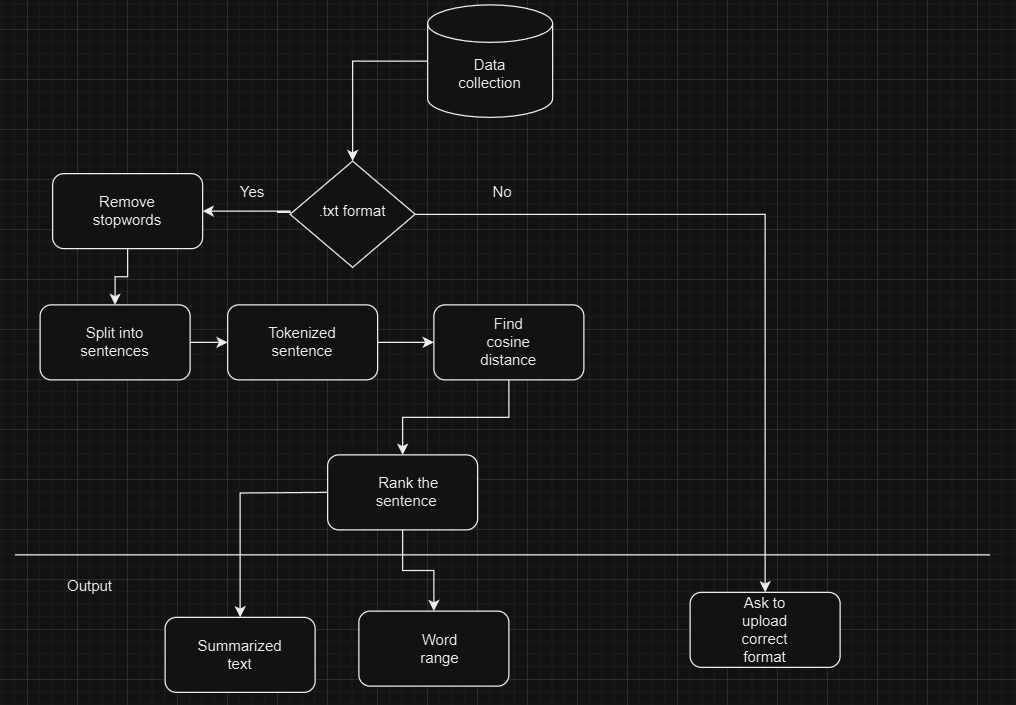
|  |  |
| --- | --- |
| **PROCESSOR** | Intel Core i5 |
| **RAM** | 4GB or above (DDR4 RAM) |
| **GPU** | Intel Integrated Graphics |
| **HARD DISK** | 6GB |
| **PROCESSOR FREQUENCY** | 1.5 GHz or above |

### SOFTWARE SPECIFICATIONS

### Python

### Streamlit

* 1. **SYSTEM DESIGN**
     1. **ARCHITECTURE DIAGRAM**

****

**Fig.1 Architecture Diagram**

The novel text summary tool BRIEFIFY was created to simplify the sometimes difficult task of distilling long publications into clear and educational summaries. Users can upload text files with ease using the user-friendly web application interface; the system ensures compatibility by only accepting files in the ".txt" format. In addition to streamlining the input process, this first step validates the file format upon submission, guaranteeing data integrity. BRIEFIFY starts its advanced processing pipeline as soon as the text file is uploaded. The material is painstakingly divided into separate sentences by the algorithm, which follows accepted formatting guidelines to guarantee precise segmentation. The next important step is tokenization, which breaks each sentence down into its individual words.

Now that the text has been suitably tokenized and split, BRIEFIFY uses cutting-edge natural language processing methods to evaluate the significance and semantic relevance of every sentence in the document. The algorithm determines the contextual relevance of statements by calculating the degree of similarity between pairs of sentences using metrics like cosine similarity. The foundation for later processing stages is laid by this step.

BRIEFIFY builds a semantic network of the document by utilizing graph theory and methods such as PageRank. The network is comprised of phrases as nodes and similarity scores as edges. Sentences are given relevance scores based on their centrality and connection within the network through iterative ranking methods. The algorithm can recognize and order the most important information in the document thanks to this hierarchical ranking.

## CHAPTER 4 PROJECT DESCRIPTION

### MODULE DESCRIPTION

#### DATA COLLECTION:

First, text data is gathered by the system from a variety of sources, such as papers, publications, and websites. The text summarizing technique uses this data, which spans a wide range of subjects and areas, as its input.

* + 1. **PREPROCESSING:**

The text data is collected and then goes through a number of preparation stages in order to get it ready for additional analysis and summary. Tokenization, which divides the text into discrete words or tokens, is one of the first stages. This procedure is essential since it aids in determining the text's fundamental building blocks, which facilitates analysis and manipulation. Sentence splitting is also done, breaking the content up into separate sentences. This stage is crucial for comprehending the text's organization and for later procedures that depend on sentence-level analysis.

* + 1. **SIMILARITY MATCHING:**

Using cosine distance or other similarity metrics, the algorithm determines how similar the sentences are to one another in the text. In this step, each sentence's significance and relevance within the whole document's context are assessed.

**4.1.4 RANKING AND SELECTION :**

Based on their value within the content, the sentences are ranked by the system using graph-based techniques such as PageRank. In order to ensure that the most pertinent information is provided, the summary is formed from the sentences that rank highest.

* + 1. **KEYWORD GENERATION :**

Keyword generation using SpaCy involves processing text to extract significant words and phrases. First, the text is analyzed to identify named entities and noun chunks. These entities and chunks are then filtered and counted for frequency. The most common phrases are selected as keywords, representing the main topics of the text.

## CHAPTER 5 IMPLEMENTATION AND RESULTS

### IMPLEMENTATION

This text summarization web app, built with Streamlit, uses cosine similarity and PageRank algorithms to condense uploaded text files into concise summaries. Users simply upload a file, and the app intelligently extracts the most relevant sentences, providing an efficient way to digest large amounts of text.

* + 1. **Input Phase**

This phase makes it easier to obtain text data for summarization that comes from files ending in.txt. Users use the online interface to upload files, which are limited to.txt files.The system checks the file format after submission to make sure it complies.Assuming that sentences conclude with a period and a space, it divides the text into separate sentences. After that, each sentence is tokenized into words, with non-alphabetic characters removed. Ultimately, the function yields a list of tokenized sentences that are prepared for additional processing, including summarization or similarity calculations.

**5.1.2 NLTK Stopwords**

This phase ensures that only relevant textual content is processed for summarization by filtering out common English stopwords using NLTK's built-in stopwords corpus. NLTK provides a comprehensive list of stopwords, including articles, prepositions, and conjunctions, which are often not indicative of the central theme or meaning of the text.By excluding these stopwords from the analysis, the system focuses on significant words and phrases, improving the accuracy and relevance of the generated summaries. This step optimizes the summarization process by prioritizing essential content while reducing noise and irrelevant information.

* + 1. **Similarity Matrix**

Quantifying the similarity between sentence pairs is a necessary step in creating the similarity matrix for text summarization. This is the general procedure:

**1.Tokenization and Preprocessing:** Every sentence is tokenized into individual words, and stopwords and a lowercase conversion are done as part of the preprocessing step.

**2. Creating the Matrix:** Using the tokenized representations of each pair of sentences, a similarity score is calculated. This score shows the degree of content similarity between the sentences.

**3. Cosine Similarity:** When determining how similar two vectors are to one another, cosine similarity is frequently employed. The similarity between the vector representations of two sentences is calculated here.

**4. Matrix Representation:** Each cell in the matrix reflects the degree of similarity between the matching pair of sentences, and the similarity scores are kept there. The square matrix's dimensions correspond to the total number of sentences in the manuscript.

**5. PageRank Usage:** The PageRank algorithm is based on the similarity matrix. By giving the connection information between sentences, it enables PageRank to evaluate each sentence's significance in relation to other phrases.

In general, the process of creating a similarity matrix serves as the basis for locating significant sentences within the text, which makes the process of summarizing it easier.

**5.1.4 PageRank Algorithm**

The PageRank algorithm is used in text summarization to order sentences according to their significance in the content. Sentences are viewed as nodes in a network, and the similarity between sentences is represented by the connections (edges) that connect them. The creators of Google, Larry Page and Sergey Brin, created PageRank, which gives each text a score determined by how important the sentences it is connected to.

**1. Building the Network:** We start by creating a graph in which every sentence is a node and the connections (edges) between sentences are determined by how similar they are to each other.

**2. Determining Sentence Importance:** PageRank uses an iterative process to rate each sentence's importance by adding the scores of all the sentences that are connected to it. In this instance, a sentence's importance is increased in proportion to its similarity score.

**3. Convergence:** This process keeps on until the scores converge, which shows that sentences' relative importance has stabilized.

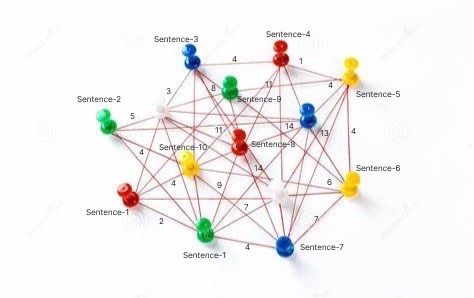
 **4. Ranking:** The final step involves assigning importance scores to each sentence, with the highest-scoring sentences being chosen to create the summary.

Fig2 Page Ranking

#### KEYWORDS USING SPACY:

1. **Text Processing**: Tokenize the input text and parse it using SpaCy.
2. **Named Entity Recognition (NER)**: Identify entities like persons, organizations, and locations in the text.
3. **Noun Chunk Extraction**: Extract meaningful phrases containing nouns and modifiers.
4. **Filtering**: Remove irrelevant or short phrases and stopwords.
5. **Frequency Counting**: Count the occurrence of remaining phrases.
6. **Keyword Selection**: Select the most frequent and meaningful phrases as keywords.
7. **Insights**: Keywords offer valuable insights into the main themes and topics of the text, aiding in summarization and content understanding.

**APPENDIX:**

**//app.py**

import streamlit as st

from model import generate\_summary, extract\_keywords

# Set page title and background color

st.set\_page\_config(page\_title="BRIEFIFY", page\_icon=":bulb:", layout="wide")

# Set title and subtitle

st.title("BRIEFIFY")

st.subheader("AI Based Text Summarizer and Keyword Extractor")

# Upload file and process

file = st.file\_uploader("Upload file", type=['txt'])

if file is not None:

st.write("File uploaded successfully!")

st.write("Generating Summary and Extracting Keywords...")

# Read file content

text = file.read().decode("utf-8")

# Extract keywords

keywords = extract\_keywords(text, top\_n=10)

# Display keywords in a box

st.subheader("Keywords Highlighted")

with st.expander("Keywords"):

for keyword in keywords:

st.markdown(f"- \*\*{keyword}\*\*")

# Generate summary

summary = generate\_summary(text, 2)

# Display number of words in the input file and summarized text

col1, col2 = st.columns(2)

with col1:

st.info("Word Count in Input File")

st.write(len(text.split()))

with col2:

st.success("Word Count in Summarized Text")

st.write(len(summary.split()))

# Display summary

st.warning("Summary")

st.write(summary)

//model.py

import nltk

from nltk.corpus import stopwords

from nltk.cluster.util import cosine\_distance

import numpy as np

import networkx as nx

import spacy

from collections import Counter

# Ensure necessary downloads

nltk.download("stopwords")

# Load spaCy's English model

nlp = spacy.load('en\_core\_web\_sm')

# Function to read the article

def read\_article(text):

article = text.split(". ")

sentences = []

for sentence in article:

sentences.append(sentence.replace("[^a-zA-Z]", " ").split(" "))

if sentences[-1] == ['']: # Remove any empty strings

sentences.pop()

return sentences

# Function to calculate sentence similarity

def sentence\_similarity(sent1, sent2, stopwords=None):

if stopwords is None:

stopwords = []

sent1 = [w.lower() for w in sent1]

sent2 = [w.lower() for w in sent2]

all\_words = list(set(sent1 + sent2))

vector1 = [0] \* len(all\_words)

vector2 = [0] \* len(all\_words)

# build the vector for the first sentence

for w in sent1:

if w in stopwords:

continue

vector1[all\_words.index(w)] += 1

# build the vector for the second sentence

for w in sent2:

if w in stopwords:

continue

vector2[all\_words.index(w)] += 1

return 1 - cosine\_distance(vector1, vector2)

# Function to build similarity matrix

def build\_similarity\_matrix(sentences, stop\_words):

# Create an empty similarity matrix

similarity\_matrix = np.zeros((len(sentences), len(sentences)))

for idx1 in range(len(sentences)):

for idx2 in range(len(sentences)):

if idx1 == idx2: #ignore if both are same sentences

continue

similarity\_matrix[idx1][idx2] = sentence\_similarity(sentences[idx1], sentences[idx2], stop\_words)

return similarity\_matrix

# Function to generate summary

def generate\_summary(text, top\_n=5):

stop\_words = stopwords.words('english')

summarize\_text = []

# Read text and split it

sentences = read\_article(text)

# Generate Similarity Matrix across sentences

sentence\_similarity\_matrix = build\_similarity\_matrix(sentences, stop\_words)

# Rank sentences in similarity matrix

sentence\_similarity\_graph = nx.from\_numpy\_array(sentence\_similarity\_matrix)

scores = nx.pagerank(sentence\_similarity\_graph)

# Sort the rank and pick top sentences

ranked\_sentence = sorted(((scores[i], s) for i, s in enumerate(sentences)), reverse=True)

for i in range(top\_n):

summarize\_text.append(" ".join(ranked\_sentence[i][1]))

# Output the summarized text

return ". ".join(summarize\_text)

# Function to read the input file

def read\_file(file\_path):

with open(file\_path, 'r', encoding='utf-8') as file:

return file.read()

# Function to extract keywords

def extract\_keywords(text, top\_n=10):

doc = nlp(text)

# Extract named entities and noun chunks

entities = [ent.text for ent in doc.ents if ent.label\_ in ['PERSON', 'ORG', 'GPE', 'LOC', 'EVENT']]

noun\_chunks = [chunk.text for chunk in doc.noun\_chunks if len(chunk) > 1]

# Combine and count the frequency of entities and noun chunks

all\_phrases = entities + noun\_chunks

frequency = Counter(all\_phrases)

# Get the most common phrases

most\_common\_phrases = frequency.most\_common(top\_n)

keywords = [phrase for phrase, count in most\_common\_phrases]

return keywords

### OUTPUT SCREENSHOTS

**5.2.1 Home Page**

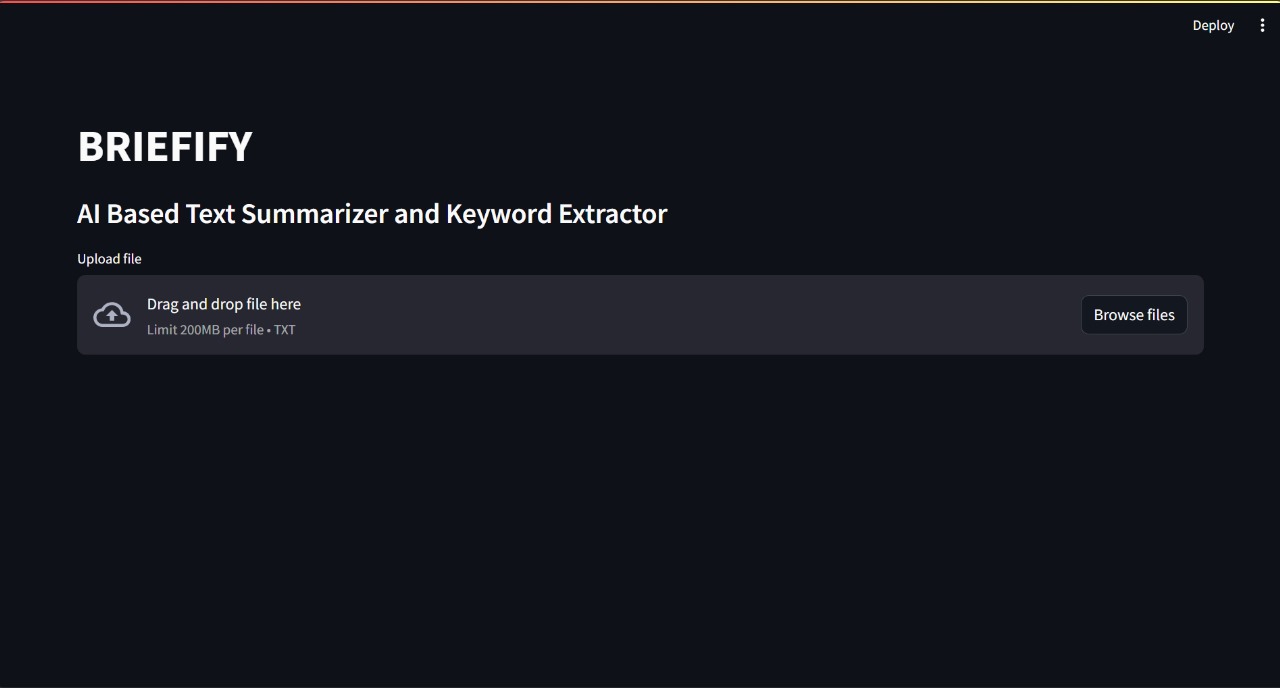


Fig 3. Shows the Home screen of the web page , here user can give their input file in .txt format.

**5.2.2 File Insertion**

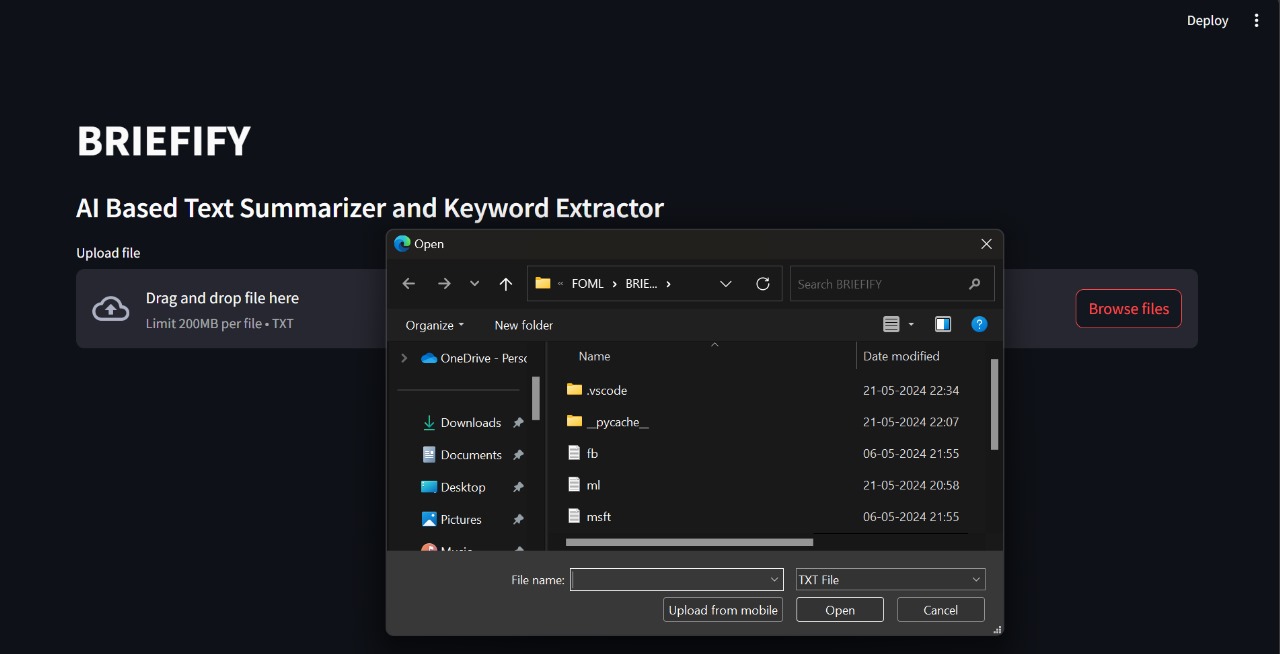


Fig 4. Shows the file insertion into the model

**5.2.3 Output Page**

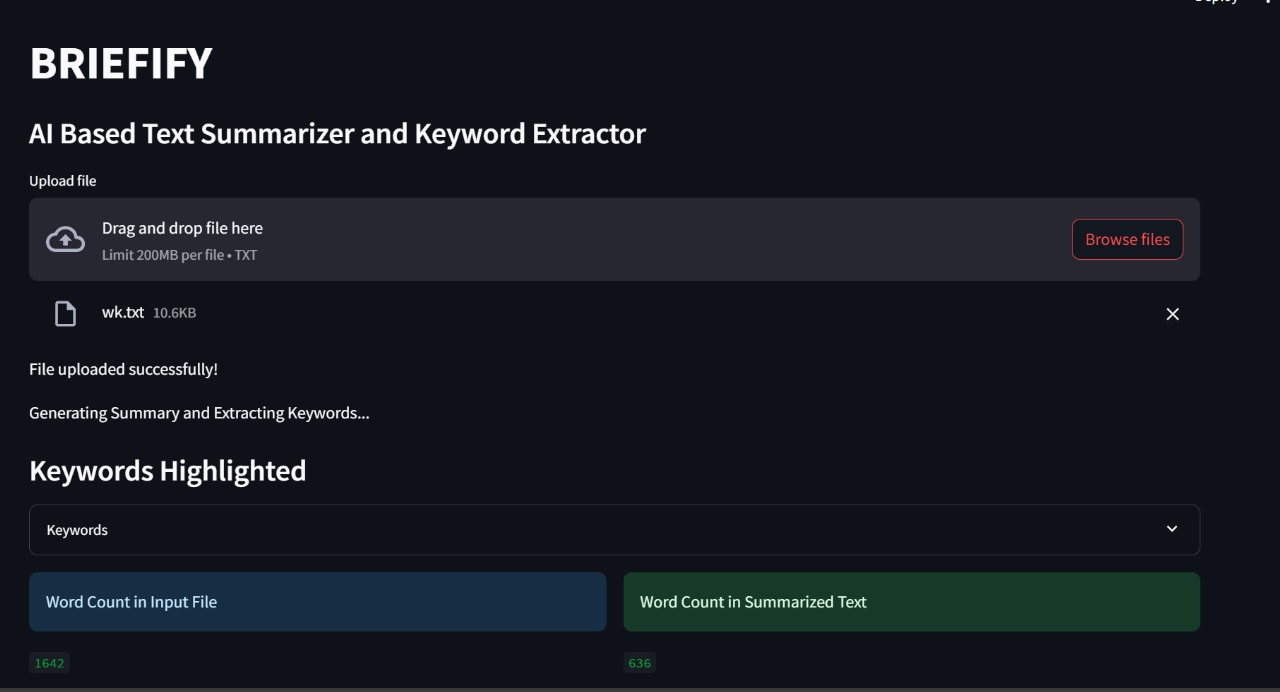


Fig5. Shows the model generated output overview , where user can see how small their input file has been summarized and an option to toggle the keywords.

**5.2.4 Keywords Page**

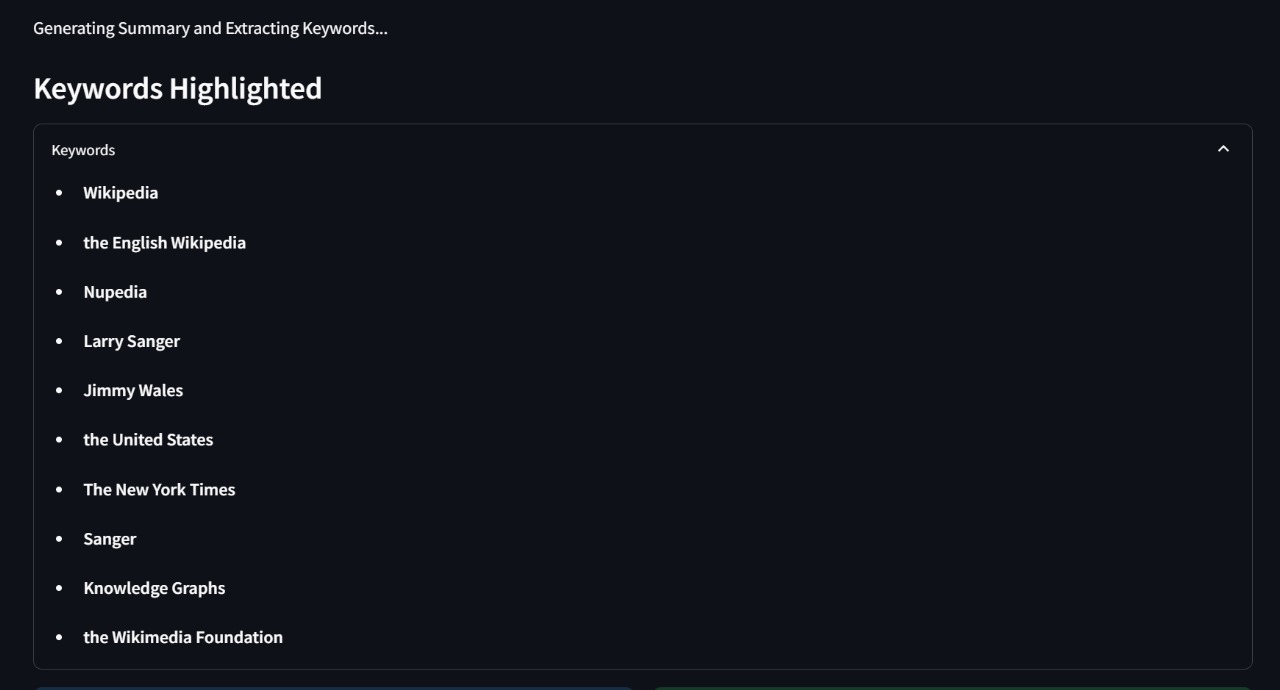
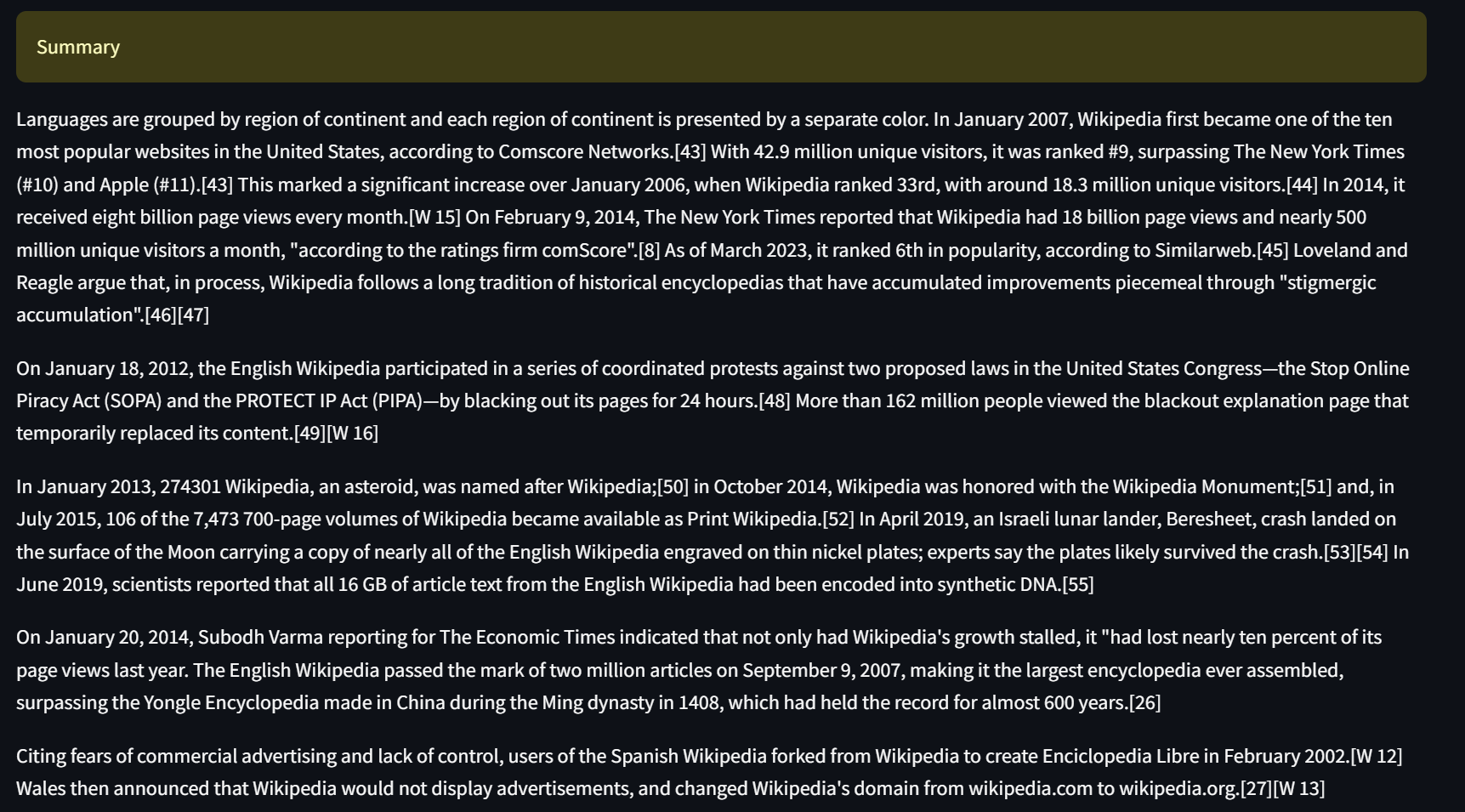


Fig 6 . shows all the keywords generated

**5.2.5 Summary Page**



## CHAPTER 6

**CONCLUSION AND FUTURE ENHANCEMENTS**

### CONCLUSION

In summary, the development and implementation of a text summarizing web application that uses PageRank and cosine similarity algorithms to condense lengthy textual content into brief summaries has been the main goal of this project. This strategy has shown to be quite successful in giving consumers a rapid and effective way to process large amounts of textual information, which makes it very useful in the information-rich environment of today. The project's preparation stage, which includes sentence splitting, tokenization, and stopword removal, makes sure the text data is ready for additional analysis. The program is able to determine the most pertinent sentences for summarizing by methodically dividing the material into digestible chunks. In order to produce high-quality summaries that preserve the spirit of the original text, this preprocessing step is essential. The application makes use of PageRank to rank sentences according to their value inside the page and cosine similarity to gauge the relevance of sentences. The program can provide brief summaries that preserve the main ideas and context of the original material because to this mix of strategies. Large-scale information management is made easier with the help of an intuitive interface that allows users to upload text files that are analyzed and summarized.

### FUTURE ENHANCEMENTS

**Interactive Visualization**: Implement interactive visualizations to display keyword frequencies, summarization results, or text analytics insights using libraries like Plotly or Bokeh.

**Customizable Summarization**: Allow users to customize summarization parameters such as the number of sentences or words in the summary, enabling tailored summarization based on their preferences.

**Multi-language Support**: Extend the project's capabilities to support multiple languages by integrating language-specific models and libraries, catering to a broader audience.

### REFERENCES

1. S. R. Rahimi, A. T. Mozhdehi and M. Abdolahi, "An overview on extractive text summarization," 2017 IEEE 4th International Conference on Knowledge-Based Engineering and Innovation (KBEI), Tehran, Iran, 2017, pp. 0054-0062, doi: 10.1109/KBEI.2017.8324874.
2. T. Islam, M. Hossain and M. F. Arefin, "Comparative Analysis of Different Text Summarization Techniques Using Enhanced Tokenization," 2021 3rd International Conference on Sustainable Technologies for Industry 4.0 (STI), Dhaka, Bangladesh, 2021, pp. 1-6, doi: 10.1109/STI53101.2021.9732589.
3. C. Lakshmi Devasena and M. Hemalatha, "Automatic Text categorization and summarization using rule reduction," IEEE-International Conference On Advances In Engineering, Science And Management (ICAESM -2012), Nagapattinam, India, 2012, pp. 594-598.
4. C. HARK, T. UÇKAN, E. SEYYARER and A. KARCI,

"Graph-Based Suggestion For Text Summarization," 2018 International Conference on Artificial Intelligence and Data Processing (IDAP), Malatya, Turkey, 2018, pp. 1-6, doi: 10.1109/IDAP.2018.8620738.

1. A. Mishra, A. Sahay, M. a. Pandey and S. S. Routaray, "News text Analysis using Text Summarization and Sentiment Analysis based on NLP," 2023 3rd International Conference on Smart Data Intelligence (ICSMDI), Trichy, India, 2023,

pp. 28-31, doi: 10.1109/ICSMDI57622.2023.00014.

1. B. Khan, Z. A. Shah, M. Usman, I. Khan and B. Niazi, "Exploring the Landscape of Automatic Text Summarization: A Comprehensive Survey," in IEEE Access, vol. 11, pp. 109819-109840, 2023, doi: 10.1109/ACCESS.2023.3322188.
2. 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.
3. P. Raundale and H. Shekhar, "Analytical study of Text Summarization Techniques," 2021 Asian Conference on Innovation in Technology (ASIANCON), PUNE, India, 2021,
4. P. Shah and N. P. Desai, "A survey of automatic text summarization techniques for Indian and foreign languages," 2016 International Conference on Electrical, Electronics, and Optimization Techniques (ICEEOT), Chennai, India, 2016,
5. R. Alfred, J. H. Obit, C. P. -Y. Chin, H. Haviluddin and Y. Lim, "Towards Paddy Rice Smart Farming: A Review on Big Data, Machine Learning, and Rice Production Tasks," in IEEE Access, vol. 9, pp. 50358-50380, 2021, doi: 10.1109/ACCESS.2021.3069449.