

# **A Comprehension Bot**



Shri Vile Parle Kelavani Mandal's

**DWARKADAS J. SANGHVI COLLEGE OF ENGINEERING**

(Autonomous College Affiliated to the University of Mumbai)

NAAC Accredited with "A" Grade (CGPA : 3.18)



## **A Comprehension Bot**

Submitted in partial fulfillment of the requirements

of the degree of

**B.Tech Computer Engineering**

By

**Rushi Desai                      60004190096**

**Saloni Patel                      60004190099**

**Shourya Kothari              60004190103**

**Guide:**

**Prof. Pankaj Sonawane**

Assistant Professor



**University of Mumbai**

**A.Y. 2022 – 2023**



Shri Vile Parle Kelavani Mandal's

**DWARKADAS J. SANGHVI COLLEGE OF ENGINEERING**

(Autonomous College Affiliated to the University of Mumbai)

NAAC Accredited with "A" Grade (CGPA : 3.18)



## **CERTIFICATE**

This is to certify that the project entitled, **“Comprehension Bot”** is a bonafide work of **“Rushi Desai (60004190096), Saloni Patel (60004190099), Shourya Kothari (60004190103)”** submitted in the partial fulfillment of the requirement for the award of the Bachelor of Technology in \_\_\_\_\_

**Prof. Pankaj Sonawane**

**Dr. Meera Narvekar**

**Head of Department**

**Dr. Hari Vasudevan**

**Principal**

**Place:**

**Date:**



Shri Vile Parle Kelavani Mandal's

**DWARKADAS J. SANGHVI COLLEGE OF ENGINEERING**

(Autonomous College Affiliated to the University of Mumbai)

NAAC Accredited with "A" Grade (CGPA : 3.18)



## CERTIFICATE

This is to certify that the project entitled, “**Comprehension Bot**” is a bonafide work of “**Rushi Desai (60004190096), Saloni Patel (60004190099), Shourya Kothari (60004190103)**” submitted in the partial fulfillment of the requirement for the award of the Bachelor of Technology in \_\_\_\_\_

**Prof. Pankaj Sonawane**

**Name of the External Guide**

**Dr. Meera Narvekar**

**Dr. Hari Vasudevan**

**Head of the Department**

**Principal**

**Place:**

**Date:**



Shri Vile Parle Kelavani Mandal's

**DWARKADAS J. SANGHVI COLLEGE OF ENGINEERING**

(Autonomous College Affiliated to the University of Mumbai)

NAAC Accredited with "A" Grade (CGPA : 3.18)



## **DECLARATION**

We declare that this written submission represents our ideas in our own words and where others ideas or words have been included, we have adequately cited and referenced the original sources. We also declare that, we have adhered to all the principles of academic honesty and integrity and have not misrepresented or fabricated or falsified any idea/data/fact/source in our submission. We understand that any violation of the above will be a cause for disciplinary action by the Institute and can also evoke penal action from the sources, which have thus not been properly cited or from whom proper permission has not been taken, when needed.

**Rushi Desai (60004190096)**

**Saloni Patel (60004190099)**

**Shourya Kothari (60004190103)**

**Place:**

**Date:**



Shri Vile Parle Kelavani Mandal's

**DWARKADAS J. SANGHVI COLLEGE OF ENGINEERING**

(Autonomous College Affiliated to the University of Mumbai)

NAAC Accredited with "A" Grade (CGPA : 3.18)



## APPROVAL SHEET

Project entitled, “**Comprehension Bot**”, submitted by “**Rushi Desai (60004190096), Saloni Patel (60004190099), Shourya Kothari (60004190103)**” is approved for the award of the Bachelor of Technology in

---

**Signature of Internal Examiner**

**Signature of External Examiner**

**Prof. Pankaj Sonawane**

**Dr. Meera Narvekar**

**Internal Guide**

**Head of the Department**

**Dr. Hari Vasudevan**

**Principal**

**Place:**

**Date:**

## **Abstract**

In order to alleviate the burden of having to sift through multiple sources to comprehend a particular topic or subject, a proposed solution is a comprehension bot that utilizes Multi-document Text Summarization (MDTS) to extract relevant information from various documents, including images, diagrams, graphs, and formulae. We introduce BERT-similarity-based Firefly (BSBF) Algorithm for Multiple Document Summarization. The goal is to present a concise and cohesive summary that retains the essential information while eliminating redundancy. To achieve this, a meta-heuristic optimization algorithm called the firefly algorithm is employed, with Topic Relevance Factor, Coherency Factor, and Readability Factor used to establish the fitness function. BERT-based similarity is used to calculate these factors, which are then input to the algorithm. The proposed algorithm's performance is evaluated using DUC 2003 and DUC 2004 datasets and compared to previous meta-heuristic and graph-based techniques.

## Contents

Chapter	Contents	Page No.
1	INTRODUCTION	1
	1.1 Description	2
	1.2 Problem Formulation	2
	1.3 Motivation	3
	1.4 Proposed Solution	3
	1.5 Scope of the project	4
2	REVIEW OF LITERATURE	5
3	SYSTEM ANALYSIS	8
	3.1 Functional Requirements	8
	3.2 Non-Functional Requirements	8
	3.3 Specific Requirements	8
4	ANALYSIS MODELING	9
	4.1 Data Modeling	9
5	DESIGN	10
	5.1 Architectural Design	10
	5.2 User Flow Diagram	11
	5.2 GUI Design	12
6	IMPLEMENTATION	13
	6.1 Algorithms/Methods Used	13
	6.2 Working Code	18
7	TESTING	20
	7.1 Test Cases	20
8	RESULTS and DISCUSSIONS	21
9	CONCLUSION and FUTURE SCOPE	25
10	REFERENCES	26



## **List of Figures**

<b>Fig. No.</b>	<b>Figure Caption</b>	<b>Page No.</b>
1	Functional Model DFD Level 0	9
2	Functional Model DFD Level 1	9
3	Architectural Diagram	10
4	User Flow Diagram	11
5	User Interface snippet	12
6	Drop multiple PDF files	18
7	Output summary with accuracy scores	19
8	Rouge Score Comparison Of BFBS Algorithm With Other Methods On Duc-2003 Dataset.	23
9	Rouge Score Comparison Of BFBS Algorithm With Other Methods On Duc-2003 Dataset.	24

## **List of Tables**

<b>Table No.</b>	<b>Table Caption</b>	<b>Page No.</b>
1	Description of Dataset	21
2	Comparison of BSBF with Other Methods on Duc-2003 Dataset	22
3	Comparison of BSBF with Other Methods on Duc-2004 Dataset	23

# 1. INTRODUCTION

A concise representation of text that includes a substantial portion of the content from one or multiple source texts, but is limited to no more than half the length of the original material, is commonly known as a Multi-Document Text Summarization (MDTS). A text summary's main advantage is that it can reduce the user's reading time.

Automatic text summarization can be classified into two main categories: single-document summarization and multi-document summarization. Multi-document summarising entails processing a connected collection of documents, whereas single-document summation [1] processes just one input document to produce a summary. In both cases, the summary can take the form of an abstract or an extract. An extract is an overview that is produced by excluding major portions of the input, whereas an abstract is produced by restating the input's key points. Two categories of text summary exist i.e informative and indicative [2]. A suggestive summary conveys to the user the text's main topic. The original content is around 5% shorter in this summary. A brief summary of the main content is provided by the informative summarising mechanism. Around 20% of the provided content is taken up by the useful summary.

An extractive summary technique involves selecting important sentences, paragraphs [3], or other components from the source material and combining them to form a condensed version. This technique heavily relies on statistical and linguistic features to determine the significance of each sentence. In contrast, abstractive summarization involves using advanced natural languages processing tools, such as grammar and lexicons, to rephrase the content of the text in a unique way.

According to the goal[4], the summarizer can be classified as generic, in which case the model treats the input impartially, or domain-specific, in which case the model uses domain data to produce an improved synopsis based on verified knowledge, or query-based, in which case the summary only contains well-known responses to natural language questions [3] about the input text.

As compared to summarising a single document, a multi-document summary presents greater difficulties. It addresses problems like redundant information in various papers, the compression of multiple documents, and the extraction of sentences quickly. These challenges are overcome utilizing statistical tools and optimization strategies. Any automatic summarizer must pay careful attention to the relevancy and redundancy of the document while summarising it.

By manually presenting and recognising the essential concepts within a lengthy publication, text summarising tools can help physicians and researchers save resources and time without having to read the entire text [5]. Text summarising first relies on frequency attributes to identify the text in documents that is most important to summarise. Afterward, a variety of algorithms and attributes have been incorporated by a number of summarization systems into the procedure of content selection.

Text summarization software has a variety of uses, including media monitoring, marketing for search engines, internal paperwork management, analysis of finances, digital marketing, and aiding the disabled.

## **1.1. Description**

We propose a method to implement an open-domain multi-document summarization. Multi-document summarization is a process to produce a single summary from a set of related documents collected from heterogeneous sources. Since the documents may contain redundant information, the performance of a multi-document summarization system heavily depends on the sentence similarity measure used for removing redundant sentences from the summary. We will map the syllabus uploaded by students to each topic in the books, so as to make it easier for the students to study for exams by saving time.

## **1.2. Problem Formulation**

To gain a thorough understanding of a given topic or subject, students must refer to several textbooks, reference books, websites, and other sources. In order to provide a condensed form of the information gathered, we develop a comprehension bot that accepts several files as input, extracts the relevant information from the documents, including photos, diagrams, graphs, and formulas, and then merges them.

For specific coursework, a student must refer to books from several publishers, carefully reading each one to extract only the necessary and relevant information. Making a compact version of numerous journals, research papers, and books becomes challenging for research-minded

students while incorporating novel concepts. Simply combined PDFs contain inessential data and lack images, diagrams, or any visualizations.

### **1.3. Motivation**

Students have to refer to numerous textbooks, reference books, websites, and sources to grasp in-depth knowledge of a particular topic or subject. We plan to implement a bot that can read multiple books(pdf), form connections between them, and combine two (or more) books into one while retaining all essential parts without having a huge page count.

With such a big amount of data circulating in the digital space, there is a need to develop machine learning algorithms that can automatically shorten longer texts and deliver accurate summaries that can fluently pass the intended messages.

Comprehension bot helps in reducing reading time, accelerates the process of researching for information, and increases the amount of information that can fit in an area.

### **1.4. Proposed Solution**

Typically, here is how using the extraction-based approach to summarize texts can work:

1. Introduce a method to extract the merited keyphrases from the source document. We can use part-of-speech tagging, word sequences, or other linguistic patterns to identify the keyphrases.
2. Gather text documents with positively-labeled keyphrases. The keyphrases should be compatible with the stipulated extraction technique. To increase accuracy, you can also create negatively-labeled keyphrases.
3. Train a binary machine learning classifier to make the text summarization. Some of the features you can use include:
  - Length of the keyphrase
  - Frequency of the keyphrase
  - The most recurring word in the keyphrase
  - Number of characters in the keyphrase

4. Calculate sentence similarity between sentences to remove redundant information.
5. Finally, in the test phrase, create all the keyphrase words and sentences and carry out classification for them.

### **1.5. Scope of the project**

A comprehensive bot that can read PDF books and respond to questions based on them. Additionally, the bot should be able to read a variety of books, make connections between them, and combine two or more texts. It should be brief enough to include all necessary details and cut out redundant information from the collected data. In addition to text, the bot should be able to produce charts, graphs, and formulas. Students can better understand their courses by using mappings between the submitted syllabus and the combined PDF copy.

## 2. REVIEW OF LITERATURE

In most real-world problems, an enhanced methodology for computing sentence similarity aiming for improving multi-document summarization performance is required. [6] paper have designed a new similarity measure that combines the single term-based similarity measure and an optimal local alignment-based similarity measure which takes into account the local context and the word order both when comparing two sentences.

Lavanya S. [7] proposes a question-answering system in which it must select the best answer from all feasible spans in the passage, instead of a list of answers for each question indicating they must cope with a large number of possibilities. The goal is to find the text as well as the context for any new questions that have been addressed. [7] used the bag of words method, which averaged the vectors of all the words in a sentence but did not consider the order of words. A sentence embedding method- Infsent that generates semantic sentence representations is presented. [7] is based on natural language inference data and can perform a variety of tasks. It is a method for constructing semantic sentence representations through the use of sentence embeddings. It is based on natural language inference data and can perform various tasks. Each question is either answered with a text fragment or span from the relevant reading material, or the question is unanswerable.

Moratanch, N., & Chitrakala, S. (2017) [8] proposes a method called Latent Semantic Analysis which is used for extracting hidden semantic structures of sentences and words commonly employed in text summarization tasks. It is an unsupervised learning method that does not require any external or training knowledge and extracts information from the input document's text. The presence of a large number of common words among the sentences indicates that the sentences are semantically related. [8] paper demonstrated various mechanisms of the extractive text summarization process. The process of extractive summarization is highly coherent, less redundant, and cohesive (summary and information-rich).

Oguzhan Tas and Farzad Kiyani [9] conducted in-depth research to determine how text summarization has evolved into a crucial tool for interpreting text information. Text summarization's primary goal is to condense the original text into a smaller version while retaining its informational value and overall message. Through these experiments, the authors found that text summarization methods can be classified into extractive and abstractive summarization. They came to the conclusion that an [9] extractive summary is a selection of important sentences from the original text while an abstractive overview is a method for novel phrasing describing the content of the text which requires heavy machinery from natural language processing.

K. Sarkar et al (2015) [10] presents an enhanced method for computing sentence similarity aiming for improving multi-document summarization performance. They have designed a new similarity

measure that combines the single term-based similarity measure and an optimal local alignment-based similarity measure which takes into account the local context and the word order both when comparing two sentences.

K. Sarkar [1] proposes a hybrid method that combines internal methods and external methods for improving text summarization performance. In our proposed approach, after segmenting the document into a collection of sentences, each sentence is assigned a combined score which is the weighted sum of two different scores: the score computed using internal information and the score computed using external information. A summary is generated by choosing the top few sentences from the list of ranked sentences.

The basic steps involved in MDTs are compilation and reduction of multiple documents, sentence extraction, removal of redundant information, sentence ranking or selection, and sentence ordering. Statistical tools have been used to resolve these issues previously [11][12], but the performance of such tools has been poor. Early attempts in abstract summarization used variables like key phrases, word frequency (tf-idf), and sentence placements to score sentences [12][13][14] and then apply the concept of compression-ratio to select the top  $n$  redundant sentences.

Redundancy removal is a crucial aspect of Multi-Document Text Summarization (MDTS). Several research papers, including [15] and [16], employ the maximal margin reduction (MMR) technique [17] to select the most important sentences and minimize redundancy while generating summaries. Another approach, presented in [26], involves creating a unified document from multiple sources using a combination of Google-based similarity algorithms and word embedding. The aptness issue is then addressed as an optimization problem, and the Shark Smell Optimization Algorithm [21] is utilized to handle it.

Approaches based on clustering have been implemented to ensure adequate coverage and prevent redundancy. These approaches utilize clustering techniques to group similar sentences into clusters to discover common informational topics before choosing individual sentences from each category to produce a summary [18][19][20]. Here, the use of the sentence similarity measure has a significant impact on cluster quality. Clustering-based methods are not as accurate as supervised methods that use annotated data since they rely on heuristics and assumptions about similarity. They may not be effective at capturing the overall coherence and meaning of the source documents, since they only consider sentence-level similarity.

The use of graph-based approaches has been demonstrated in numerous papers[27][28][29][30][31][32], where documents are represented in a graph with a weighted layout where every node is a phrase and the weighted edges show how similar the sentences are to one another. They rely on the overall relationships between sentences in the graph, rather than just looking at each sentence individually. Cosine similarity is a common technique used by these methods to determine the relationships between sentences. For instance, [33] combines



centrality-based and centroid-based methods to assess the similarity and significance of each sentence. The word graph produced by [34] is based on the alignment data between pairs of related phrases and uses SBERT[35] to transform each sentence into fixed vectors. This method makes it easier to construct sentences that include numerous bits of information. The grammatical correctness and informativeness of the sentences created are assessed using an intensification function and sentence scoring tool. Finally, to guarantee that the final summary only contains the highest-scoring sentences, integer linear programming is used during the sentence selection procedure.

There have been a few metaheuristic optimization algorithms that have been used for single and MDTS tasks like Shark Smell Optimisation (SSO) [21], Genetic Algorithm [22], and Cuckoo Search Optimisation [23]. These techniques prioritized minimized redundancy and extensive coverage.

### **3. SYSTEM ANALYSIS**

#### **3.1. Functional Requirements**

- Extract data from the documents.
- Compilation of the collected data.
- Summarization of the text ensuring minimum redundancy and maximum relevance

#### **3.2. Non-Functional Requirements**

- High reliability and accuracy.
- Minimum redundancy and Word Error Rate (WER)
- OS independent
- High Rouge score

#### **3.3. Specific Requirements**

- Software:
  - Python 3.7+
  - Pytorch
  - Open.ai gym
  - Numpy
  - Pandas
  - NLTK
  - Sentencepiece
  - huggingface
  - It will be platform independent and hence can be run on any operating system
  - Colab workbook
- Hardware
  - Any 64-bit processor with any OS installed
  - At least 4 GB RAM
  - At least 2 GB storage capacity available

## 4. ANALYSIS MODELING

### 4.1. Functional Modeling (DFDs *with specifications*) *mandatory for all projects*

#### DFD LEVEL 0

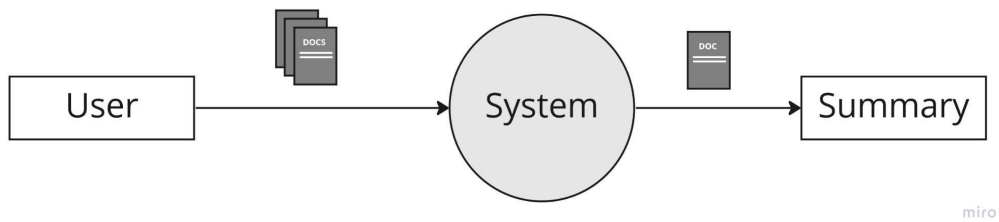


Fig.1 Functional Model DFD Level 0

#### DFD LEVEL 1

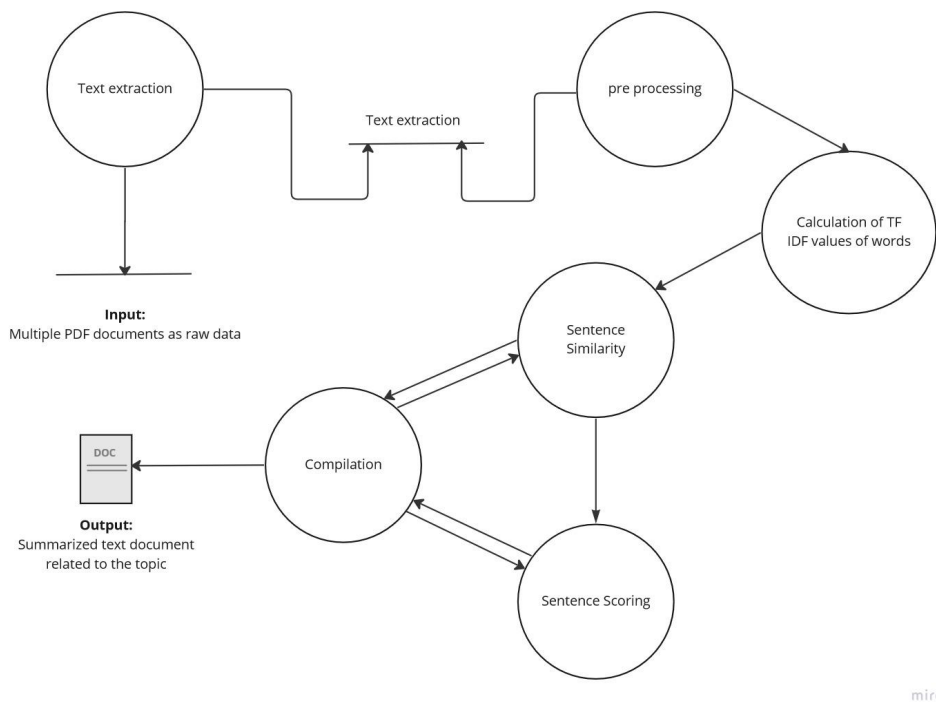
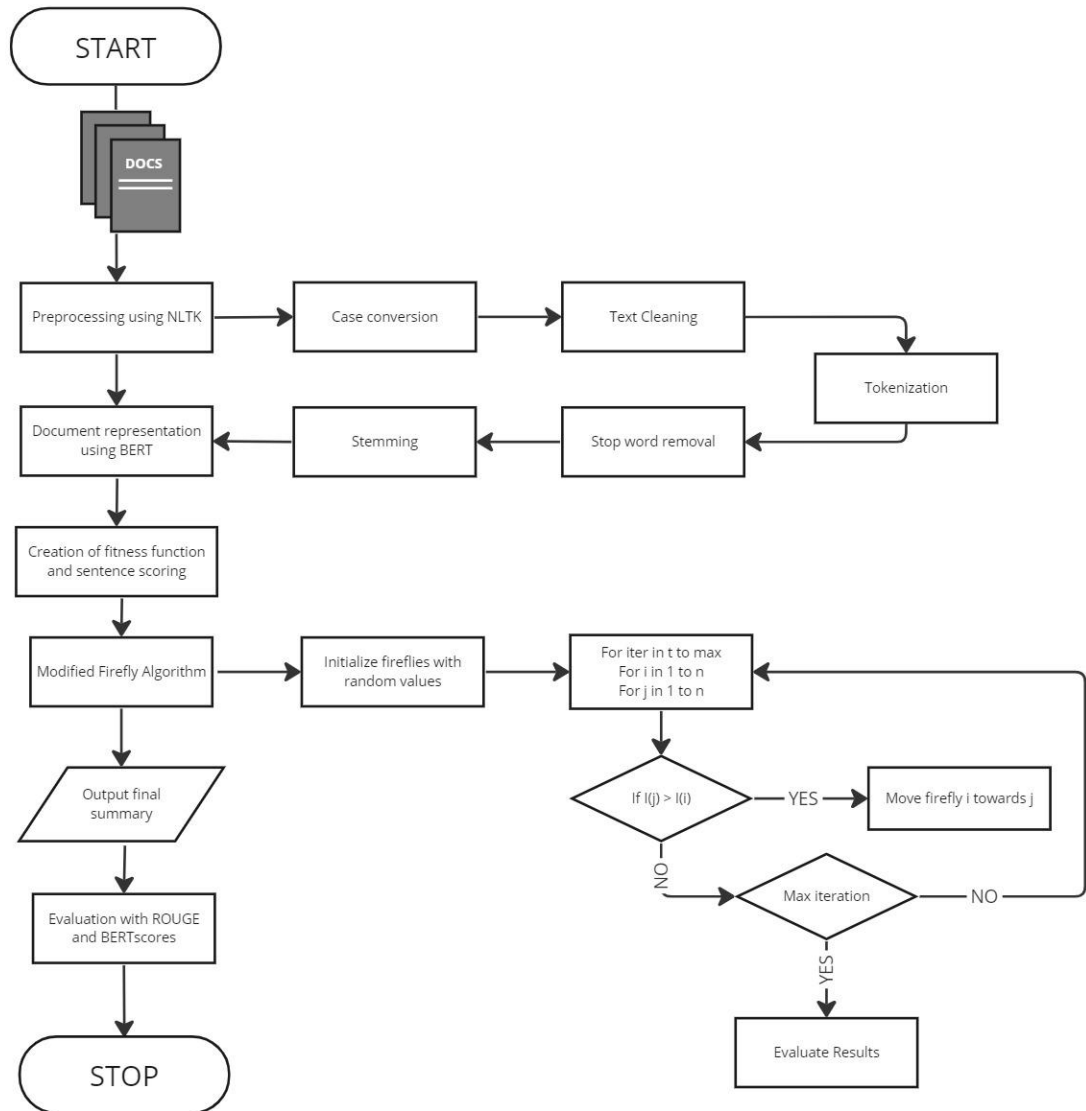


Fig.2 Functional Model DFD Level 1

## 5. DESIGN

### 5.1. Architectural Design (*Project Flow /architecture with description*)



miro

Fig.3 Architectural Design Diagram

## 5.2 User Flow diagram

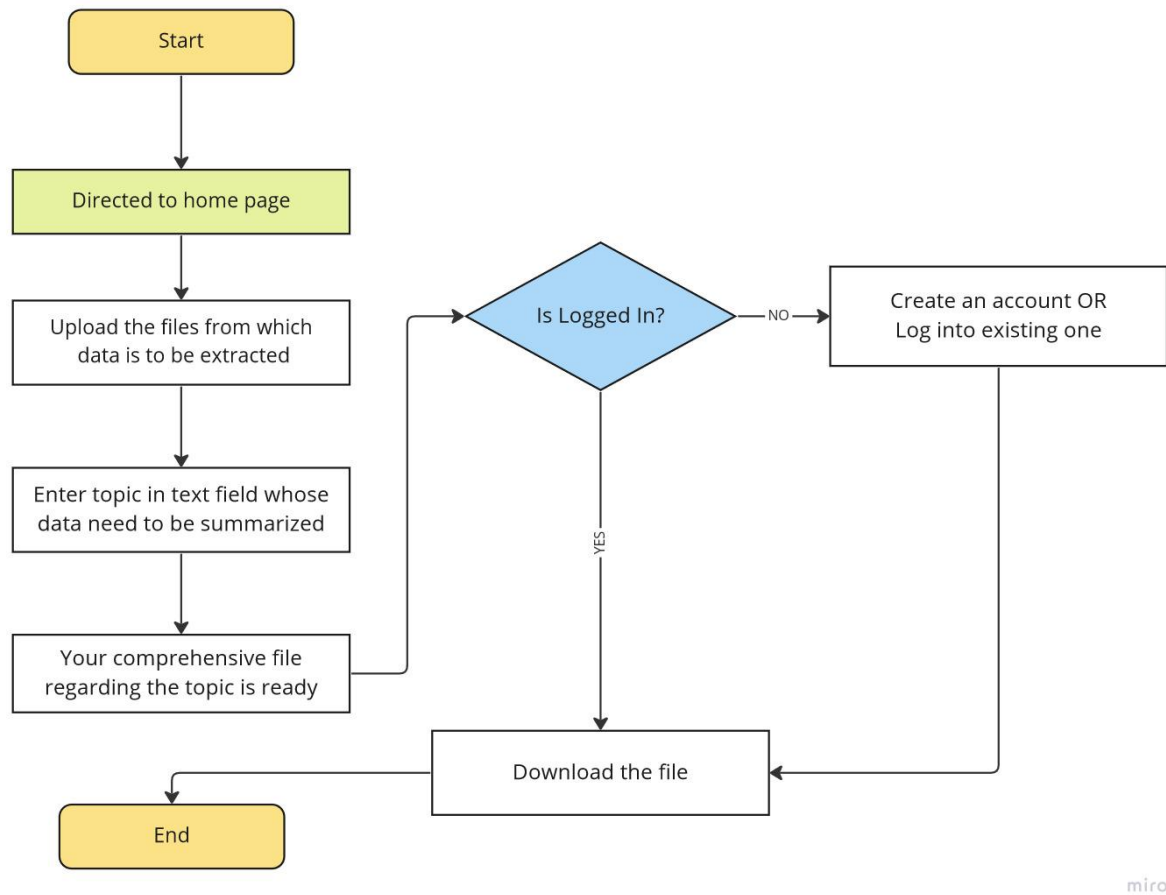


Fig.4 User Flow Diagram

## 5.3 Graphical User Interface (GUI Design)

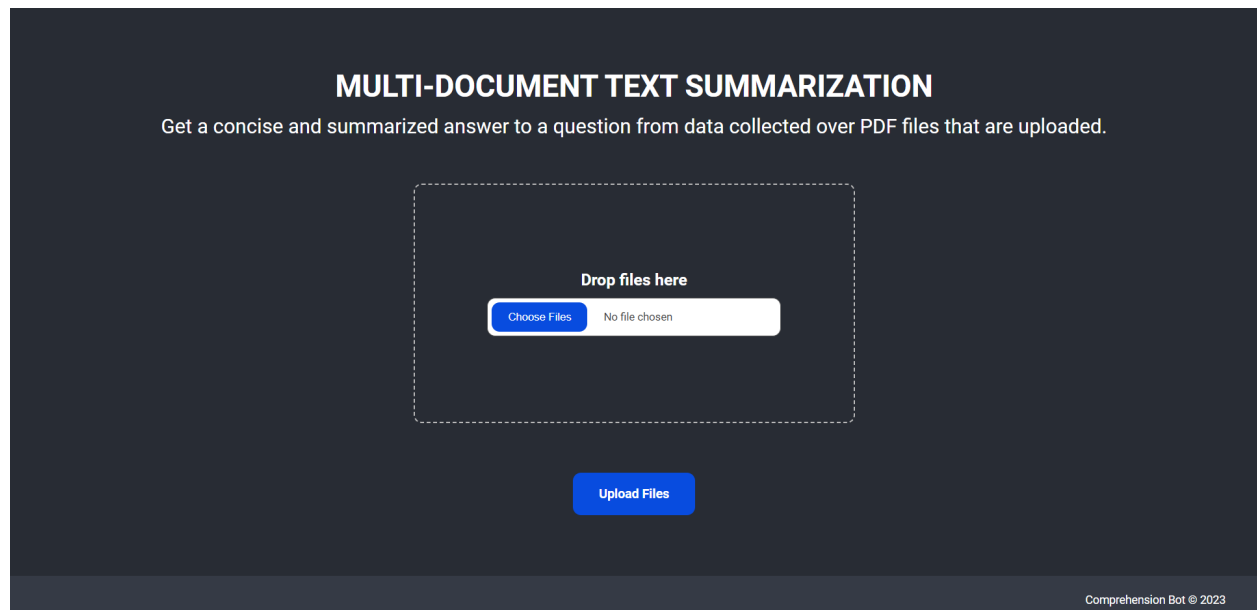


Fig.5 User Interface Snippet

## **6. IMPLEMENTATION**

### **6.1. Algorithms/Methods Used**

In the literature survey, we saw meta-heuristic optimization algorithms being used to generate multi-document summaries. One such algorithm is the firefly algorithm [36], which draws inspiration from nature, specifically from a swarm of fireflies. The method is based on how fireflies behave, where each one is drawn to the firefly which is the brightest nearby. A version of this algorithm has been used for MTDS [37]. We propose a similar algorithm with a difference in document representation and calculation of fitness score.

### **6.2 Preprocessing**

Preprocessing is an essential step in any NLP task that involves transforming the raw input documents into a format easily processed by the summarization algorithm.

#### **6.2.1 Case conversion**

All text is converted to lowercase to ensure uniformity during processing.

#### **6.2.2 Text cleaning**

The raw text may contain noise, irrelevant information, or formatting tags. Text cleaning involves removing these elements from the text, such as HTML tags, special characters, and punctuation.

#### **6.2.3 Sentence Segmentation**

Once the documents are selected and cleaned, the text must be segmented into individual sentences. This is done by identifying sentence boundaries based on punctuation, capitalization, and other linguistic cues.

#### **6.2.4 Sentence Normalization or Stop Word Filtering**

Normalization involves converting words to their base form and removing common words that do not add meaning or are not important, such as "the" and "a."

#### **6.2.5 Stemming**

Stemming returns inflected (or occasionally derived) words to their root form. Using singular rather than plural nouns, for instance, or dropping the -ed or -ing from verbs are other examples. Many NLP tools use stemmer algorithms to carry out the stemming process.

### **6.3 Document Representation**

We represent the document with BERT (Bidirectional Encoder Representations from Transformers) [38]. BERT is a bi-directional transformer model pre-trained on 3300M words for Masked Language Modelling (MLM) tasks. BERT uses an encoder model to create word embeddings in a latent space. We use 12 transformer layers to get a representation of the document. The final hidden layer's output serves as the embedding which is later used to calculate similarity in the fitness function.

When BERT is used for document representation, the document is first tokenized into a sequence of words, and then each word is converted into an embedding using BERT's pre-trained word embeddings. The embeddings are then passed through a pre-trained BERT model, which produces a contextualized representation of each word in the document.

These contextualized representations are then combined in a variety of ways to produce a single representation of the entire document. One common approach is to take the mean or max pooling of the contextualized word embeddings to obtain a fixed-length vector representation of the document.

The resulting document representation can be used for a variety of downstream tasks, such as document classification, information retrieval, and document summarization. BERT's ability to capture complex contextual relationships between words and phrases makes it a powerful tool for document representation.

### **6.4 Fitness Function**

A summary is said to be good when it accurately and concisely conveys vital information from the original text in an articulate way. A good summary includes the main points of the text while omitting minor details, examples, and supporting evidence. A well-written summary should have coherence and flow, meaning each sentence should be logically connected to the previous and next sentences. This creates a sense of continuity and makes the summary more readable and understandable. To achieve these features we consider three factors as mentioned in [37] as well: i) Topic Relativity factor ii) Linking Factor and iii) Readability factor.

#### **6.4.1 TOPIC RELATIVITY FACTOR (TRF)**

The Topic Relativity Factor (TRF) is a measure of the degree to which an abstract is connected to its subject matter. It is calculated by considering two factors: (i) the resemblance between the



title and summary, referred to as Title Similarity (TS), and (ii) the similarity of the summary to the original text, known as Original Text Similarity (OTS).

To calculate the degree of similarity between the summary and the title, the average similarity between each sentence and the title is first calculated. This value is then normalized by dividing it by the maximum similarity.

$$TS = \frac{\sum bertsim(S_j, q)}{S} \quad (1)$$

$S_j$  stands for the summary's  $j$ th sentence, while  $q$  stands for the title.  $S$  stands for the summary's total number of sentences. To determine the similarity between each sentence in the summary and the text, we employ the BERT-similarity metric. This approach has several advantages, as highlighted in studies such as [30] and [34].

One of the key benefits of using BERT-similarity is its ability to accurately capture the semantic similarity between different texts, even when the texts use distinct words or syntactic structures. This is explained by the fact that the BERT has already been trained on a sizable corpus and has honed its ability to represent word and phrase meanings based on context.

$$OTS = \frac{\sum bertsim(S_j, T_i)}{S * T} \quad (2)$$

Where  $T$  denotes the total number of sentences in the original text and  $T_i$  denotes the  $i$ th sentence in the original text. Finally, we add  $TS$  and  $OTS$  to determine the TRF.

$$TRF = OTS + TS \quad (3)$$

#### 6.4.2 Linking Factor (LF)

Linking or cohesion refers to the degree of connectedness and unity among the elements of a text, such as words, phrases, and sentences. It reflects how well these different parts are integrated into a meaningful and coherent whole. The summary is converted into a weighted graph, where each sentence is represented as a node and the edge weight denotes the degree of similarity between two sentences, in order to calculate the cohesiveness factor. Two factors are required to calculate the linking factor:  $C$  and  $M$ .

$C$  is obtained by calculating the mean of the similarities between all sentences in the summary.

$$C = \frac{\sum_{\forall S_i S_j \in \text{subsummary graph}} W(S_i S_j)}{N_s} \quad (4)$$

Where  $N_s$  is the total number of edges in the summary subgraph and  $W(S_i S_j)$  indicates the sum of the weights of all the edges on the path from sentence  $i$  to sentence  $j$ .

$$N_s = \frac{S^*(S-1)}{2} \quad (5)$$

In the summary subgraph,  $M$  is the highest weight.

$$M = \max_{i,j} \text{sim}_{i,j} \quad i, j \leq N \quad (6)$$

Finally,  $CF$  is given by the logarithm equation as mentioned in [40]. When the maximum is significantly bigger than the mean, the logarithm helps prevent the low magnitude of  $LF$ .

$$LF = \frac{\log_{10}(C^*9 + 1)}{\log_{10}(M^*9 + 1)} \quad (7)$$

#### 6.4.3 Readability Factor

A readable summary is where two sentences are correlated and there is a similarity between two consecutive sentences. However, we need to ensure that the sentences are not very similar as they may repeat the same information and increase redundancy. Hence, first, we set an upper limit between the similarity of two sentences,  $\alpha$ .

$$\text{sim}_{i,j} \leq \alpha \quad (8)$$

After experimenting with different values of  $\alpha$ , the optimal value came out to be 0.88

$RF$  is calculated as follows:

$$\text{Readability}(R) = \sum W(S_i, S_j) \quad (9)$$

$$RF = \frac{R}{\max_{\forall R_i} R_i} \quad (10)$$

Where  $R_i$  is the greatest distance in the weight matrix with a specific number of nodes.

#### 6.4.4 The Fitness Function

The aforementioned factors are used for the calculation of the fitness function. Each factor is assigned a weight that is user-defined. So for example, if a user wants a more cohesive summary, he/she can increase the weight associated with  $CF$ .

$$F = \alpha * TRF + \beta * LF + \gamma * RF \quad (11)$$

## 6.5 Firefly Algorithm

The metaheuristic optimisation technique known as the "firefly algorithm"[36] was modelled after the natural behaviour of fireflies. Fireflies are known for their ability to produce light and use it for communication, attraction, and mate selection. The algorithm was first brought to light by Xin-She Yang in 2008. The firefly algorithm is a swarm-based algorithm that simulates the behavior of fireflies.

In multi-document summarization, the goal is to generate a summary that captures the most important information from a collection of documents. The Firefly Algorithm can be used to optimize the summary by adjusting the weights assigned to each sentence in the summary.

The algorithm works by treating each sentence in the summary as a firefly, and the quality of the summary as the brightness of each firefly. The algorithm then simulates the flashing behavior of fireflies to move the fireflies toward a better solution. In this case, the better solution is one that maximizes the quality of the summary.

### 6.5.1 Representation of Firefly

Every document (firefly) is represented as a vector, and each sentence is labeled as 1 or 0 depending on whether it should be included in the summary. The method initialises K fireflies at random and determines each firefly's light intensity using a fitness function that integrates the three features stated above. The firefly with the highest fitness value is deemed the brightest firefly and serves as the leader. Other fireflies are attracted to this leader. The primary objective of this algorithm is to locate the summary that maximizes the fitness function.

### 6.5.2 Firefly Algorithm Usage

Based on the brightness of other fireflies in the population, the firefly algorithm modifies each firefly's position.

$$x_i = x_i + \beta_0 e^{-\gamma r_{ij}^2} (x_j - x_i) + \alpha(\epsilon() - 0.5) \quad (12)$$

(Eq.12) includes several parameters. The firefly's absolute brightness is represented by the value  $\beta_0$ , which is normally set to 1. A firefly's level of random movement is controlled by the parameter  $\alpha$  while the distance between them is adjusted by the parameter  $\gamma$ . The symbol for a random vector, which has values between 0 and 1, is denoted by  $\epsilon()$ . For the brightest firefly, the update equation is simplified (Eq.13) by neglecting the second term. This means that the brightest firefly does not move toward other fireflies, but rather moves randomly.

### 6.5.3 Firefly Calculations

To create a summary representation, to convert the continuous values obtained from the fitness function to binary values, any value greater than 0.5 is treated as 1, indicating that the

corresponding sentence should be included in the summary. Conversely, any value less than 0.5 is treated as 0, indicating that the corresponding sentence should not be included in the summary. This process is used to compute intermediate summaries during the summarization process, which are then further processed to generate a final summary.

It's important to note that the final summary may not adhere strictly to this binary classification process. Post-processing techniques are often used to refine the summary and ensure that it reads fluently and coherently. Additionally, some summarization techniques may use other methods to generate the final summary, such as selecting the most important sentences based on a ranking score.

#### 6.5.4 Generation of Summary

Once the maximum number of iterations is reached, the algorithm produces a set of potential summaries, each generated by a different firefly. The fitness value of each firefly is a measure of the quality of the corresponding summary.

To generate the final summary, the firefly with the highest fitness value is selected as the best firefly. The sentences with the highest values in the best firefly are then selected to form the final summary. The number of sentences selected may be limited by a maximum number of phrases or words, depending on the requirements of the task.

If multiple input documents are being summarized, the procedure continues until the maximum number of phrases or words is not achieved for all the documents. The final summary is then prepared by combining the selected sentences from all the documents.

## 6.6 Working Code

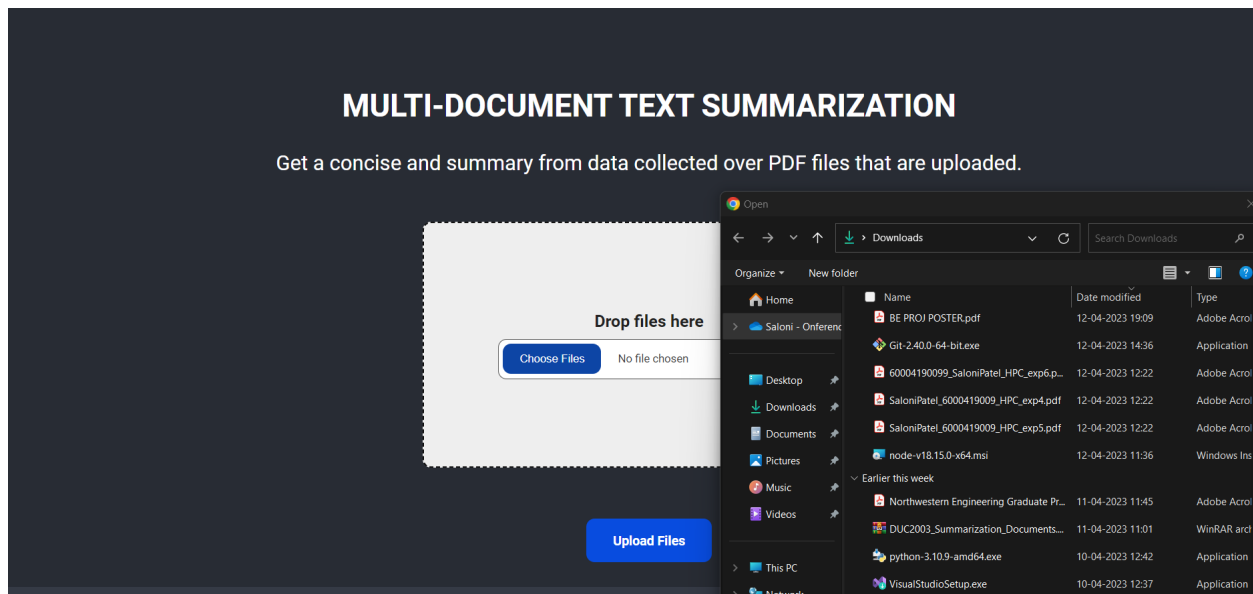


Fig.6 Drop multiple PDF files

Summary	designed to be resistant to linear and differential cryptanalysis, which are common attacks against encryption algorithms. The ShiftRows operation, which is the second operation in each round, cyclically shifts the rows of the block to the left. The first row is not shifted, the second row is shifted by one byte to the left, the third row is shifted by two bytes to the left, and the fourth row is shifted by three bytes to the left. This operation provides diffusion of the input data, which enhances the security of the algorithm. The MixColumns operation, which is the third operation in each round, mixes the columns of the block using a fixed matrix multiplication. The matrix used in the multiplication is designed to provide diffusion and confusion of the input data, which enhances the security of the algorithm. The PDF from Purdue University provides a detailed explanation of the matrix used in the MixColumns operation. The AddRoundKey operation, which is the fourth and final operation in each round, adds the round key to the block. The round key is derived from the original key using a key schedule algorithm, which generates a series of subkeys that are used in each round of the algorithm. The key schedule algorithm is designed to enhance the security of the algorithm by generating subkeys that are dependent on the original key and the number of rounds in the algorithm. The PDF from Washington University provides a more general overview of the AES algorithm. It explains the design goals of the algorithm, which include security, efficiency, and flexibility. It also explains the key sizes and block sizes used in the algorithm, and how they are related to the number of rounds in the algorithm. The PDF from IIT Kharagpur provides a detailed explanation of the key schedule algorithm used in the AES algorithm. The key schedule algorithm is used to generate a series of subkeys that are used in each round of the algorithm. The algorithm uses a combination of circular shifts, S-box substitutions, and XOR operations to generate the subkeys. The PDF provides a detailed explanation of the key schedule algorithm and how it is used to generate the subkeys. In conclusion, the AES algorithm is a symmetric encryption algorithm that uses a fixed block size of 128 bits and a key size of 128, 192, or 256 bits. The algorithm consists of several rounds, each consisting of four operations
Rouge-1 F1 Score	0.79625
Rouge-1 Recall Score	0.83621
Rouge-1 Precision Score	0.93147
Rouge-2 F1 Score	0.71263
Rouge-2 Recall Score	0.79631
Rouge-2 Precision Score	0.92148

Fig.7 Output summary with accuracy scores

## **7. TESTING**

### **7.1 Test Cases**

1. Input PDF documents with different lengths and structures.
2. PDF documents with different file sizes.
3. PDF documents with different number of pages.
4. PDF documents with similar content or redundant information.
5. PDF documents with tables, charts, and graphs.
6. PDF documents with images and diagrams.
7. PDF documents with different fonts and formatting styles.
8. PDF documents in different languages.
9. PDF documents with technical jargon and industry-specific terms

### **7.2 Type of Testing Used**

We can combine black-box and white-box testing to evaluate the multi-text document summarizer.

The effectiveness and usability of the summarizer are assessed via black-box testing from the user's perspective. In order to assess the summarizer's capacity to merge various PDF documents on a single topic and omit superfluous material while preserving the crucial content, various test cases can be applied to the summarizer.

The internal workings of the summarizer, including the methods used to find and remove duplicate information, as well as the precision of the summary procedure, can be assessed using white-box testing. This shall entail examining the summarizer's source code and evaluating its many parts and features to ensure they operate as intended.

Additionally, a number of testing approaches, including unit tests, test integration, and testing of systems, are used while testing the summarizer's many functionalities, such as file upload, document parsing, summarising, and output production. Performing both positive and negative testing helps to see how the summarizer responds to various scenarios and error conditions.

## 8. RESULTS and DISCUSSIONS

### 8.1 Dataset:

DUC (Document Understanding Conference) is a series of conferences focusing on summarization and other natural language processing tasks. DUC 2003 and DUC 2004 were two such conferences, and they included datasets for document summarization tasks.

The DUC 2003 dataset included 50 news articles with corresponding human-generated summaries. The articles covered various topics and were selected from major news sources such as the New York Times, the Associated Press, and the BBC.

The DUC 2004 dataset was similar in structure to the DUC 2003 dataset. It also included 50 news articles with corresponding human-generated summaries, and the articles covered a variety of topics and were selected from major news sources.

Both of these datasets were widely used in research on automatic summarization and were instrumental in advancing the state of the art in the field. They continue to be used as benchmark datasets for evaluating new summarization systems.

Table 1. Description of Dataset

<b>Dataset Description</b>	<b>DUC 2003</b>	<b>DUC 2004</b>
Set of Documents	30	100
Documents per set	10	10
Source documents	Associated Press (AP) newswire	English Gigaword corpus
Summary length	100	655 bytes

### 8.2 Evaluation Methodology

A generated summary or translation's resemblance to a collection of reference summaries or translations is measured using a set of metrics called ROUGE. ROUGE has several variants, including ROUGE-N (which looks at n-gram overlap), ROUGE-L, and ROUGE-W (which looks at weighted n-gram overlap) [24]. We compare only Rouge-1 and Rouge-2 scores with other existing models.

However, ROUGE has some limitations, It only looks at the surface-level overlap between the generated and reference texts and does not consider semantic similarity or other aspects of text

quality. It is sensitive to differences in the length of the generated and reference texts, which can lead to artificially high or low scores.

We deploy a new evaluation metric, the BERTScore [25], to get around these constraints. Contextual embeddings, or word representations that consider the context in which a word appears, are the foundation of the BERTScore concept. Using BERTScore, it is possible to measure semantic similarity more accurately and with less sensitivity to variations in text length.

### 8.3 Results

On the DUC-2003 dataset, BSBF achieves a Rouge-1 score of 0.4351 and Rouge-2 score of 0.1792 which is better than other algorithms used (Table 2 and Fig 8.3.1). We compare the results with Genetic algorithm with JS divergence [41], Particle Swarm Optimisation (PSO) with JS divergence [42], Ant Colony Optimisation (ACO)[43], Firefly based Text Summarization[37] and LexRank[27]. Table 2 shows the comparison of the various algorithms. We made similar comparisons on the DUC-2004 dataset, where we achieved a ROUGE-1 score of 0.4351 and a Rouge-2 score of 0.1792 as mentioned in Table 3 and shown in Fig 8.3.2.

Table 2. Comparison of BSBF with Other Methods on Duc-2003 Dataset

	<b>Evaluation Metric</b>	
<b>Methods</b>	<b>ROUGE-1</b>	<b>ROUGE-2</b>
GA(JS-Divergence)	0.4397	0.1416
PSO(JS-Divergence)	0.4321	0.1507
ACO (CosineSimilarity)	0.4231	0.1407
FbTS	0.4419	0.1602
LexRank	0.3574	0.0793
<b>BSBF</b>	<b>0.4612</b>	<b>0.1702</b>



### COMPARISON OF BSBF WITH OTHER METHODS ON DUC-2004 DATASET

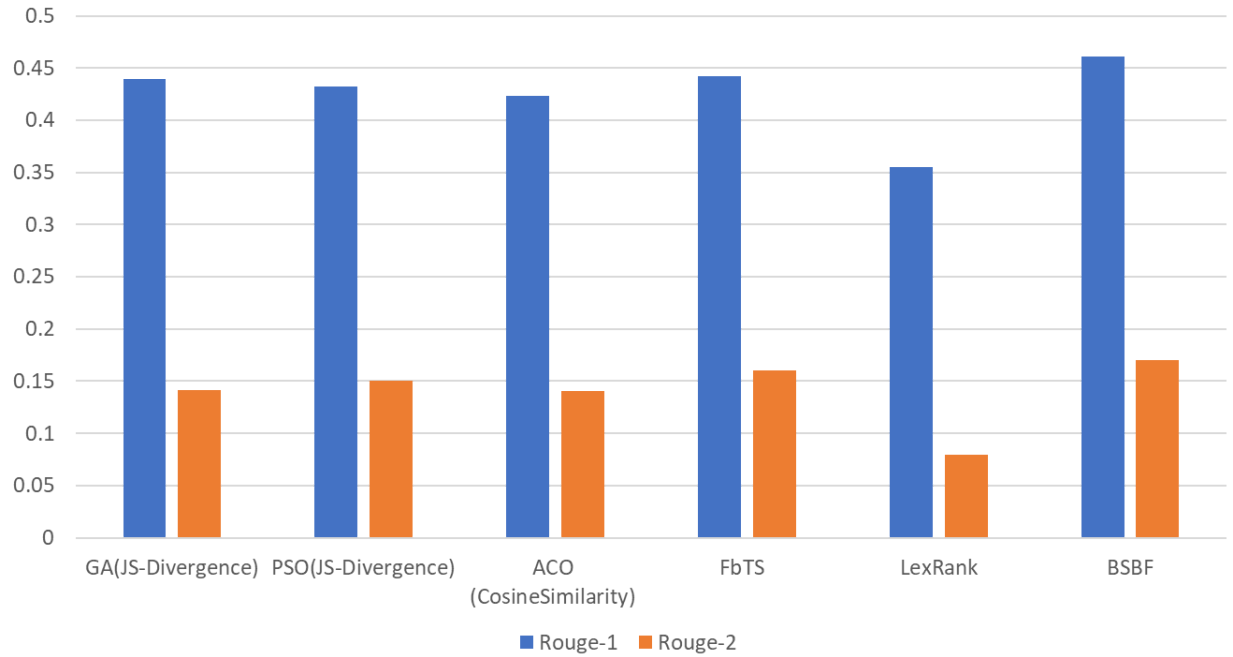


Fig.8 . Rouge Score Comparison Of Fbts Algorithm With Other Methods On Duc-2003 Dataset.

Table 3. Comparison of BSBF with Other Methods on Duc-2004 Dataset

Methods	Evaluation Metric	
	ROUGE-1	ROUGE-2
GA(JS-Divergence)	0.3546	0.0937
PSO(JS-Divergence)	0.3521	0.0935
ACO (Cosine Similarity)	0.3542	0.0837
FbTS	0.4244	0.1764
LexRank	0.326	0.079
<b>BSBF</b>	<b>0.4351</b>	<b>0.1792</b>

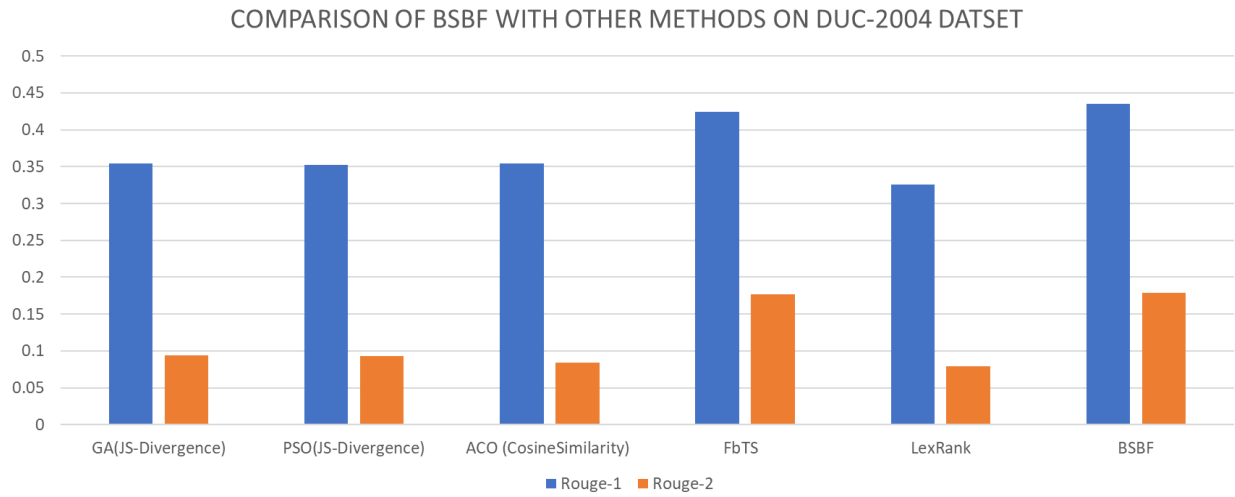


Fig 9 Rouge Score Comparison Of Fbts Algorithm With Other Methods On Duc-2004 Dataset.

## 9. CONCLUSION AND FUTURE SCOPE

We researched multiple algorithms and implementation methodologies that could be applied to summarize each document and extract relevant, non-redundant information related to each topic and compile them into one, keeping in consideration to include flowcharts, formulas, diagrams, and images. The experience of the interface is made seamless for the user to easily upload the PDF files available and then enter the topic whose data needs to be extracted. This research introduces a new algorithm for MDTS, called BSBF based on the firefly algorithm. The algorithm uses a fitness function that combines TRF, CF, and RF to rank each sentence and select the highest-scoring sentences for the summary. Experimental evaluations on DUC-2003 and DUC-2004 datasets using the ROUGE score show that the proposed algorithm outperforms genetic algorithms in terms of ROUGE-1 and ROUGE-2 scores. In order to further enhance the quality of abstractive summaries, the study recommends experimenting with new feature selection techniques and fitness features with bio-motivated algorithms and merging these extractive techniques with deep neural-based models. A summary generation may perform better when hybrid models are used. In the future scope, we aim to map the entire syllabus copy of the user rather than take the topic as input in text fields.

## REFERENCES

- [1] K. Sarkar, "Automatic Text Summarization Using Intenal and Extemal Information," 2018 Fifth International Conference on Emerging Applications of Information Technology (EAIT), Kolkata, India, 2018, pp. 1-4, doi: 10.1109/EAIT.2018.8470412.
- [2] Das, Dipanjan & Martins, André. (2007). A survey on automatic text summarization.
- [3] N. Moratanch and S. Chitrakala, "A survey on extractive text summarization," 2017 International Conference on Computer, Communication and Signal Processing (ICCCSP), Chennai, India, 2017, pp. 1-6, doi: 10.1109/ICCCSP.2017.7944061.
- [4] I. Awasthi, K. Gupta, P. S. Bhogal, S. S. Anand and P. K. Soni, "Natural Language Processing (NLP) based Text Summarization - A Survey," 2021 6th International Conference on Inventive Computation Technologies (ICICT), Coimbatore, India, 2021, pp. 1310-1317, doi: 10.1109/ICICT50816.2021.9358703.
- R. E. Sorace, V. S. Reinhardt, and S. A. Vaughn, "High-speed digital-to-RF converter," U.S. Patent 5 668 842, Sept. 16, 1997.
- [5] A. S. Almasoud, S. Ben Haj Hassine, F. N. Al-Wesabi, M. K. Nour, A. Mustafa Hilal et al., "Automated multi-document biomedical text summarization using deep learning model," Computers, Materials & Continua, vol. 71, no.3, pp. 5799–5815, 2022.
- [6] Saraf, K., & Ghosh, A. (2015). Improving graph-based multi-document text summarization using an enhanced sentence similarity measure. 2015 IEEE 2nd International Conference on Recent Trends in Information Systems (ReTIS). doi:10.1109/retis.2015.7232905
- [7] Lavanya S., End-to-End Question-Answering System Using NLP and SQuAD Dataset
- [8] Moratanch, N., & Chitrakala, S. (2017). A survey on extractive text summarization. 2017 International Conference on Computer, Communication and Signal Processing (ICCCSP). doi:10.1109/icccsp.2017.7944061.
- [9] O. Tas and F. Kiyani , "A SURVEY AUTOMATIC TEXT SUMMARIZATION", PressAcademia Procedia, vol. 5, no. 1, pp. 205-213, Jun. 2017, doi:10.17261/Pressacademia.2017.591
- [10] K. Sarkar, K. Saraf and A. Ghosh, "Improving graph based multidocument text summarization using an enhanced sentence similarity measure," 2015 IEEE 2nd International Conference on Recent Trends in Information Systems (ReTIS), 2015, pp. 359-365, doi: 10.1109/ReTIS.2015.7232905.
- [11] Gupta, Vikrant, Priyamvada Chauhan, Sohan Garg, Anita Borude and Shobha Krishnan. "An Statistical Tool for Multi-Document Summarization." (2012).
- [12] K. Sarkar, "Automatic Text Summarization Using Intenal and Extemal Information," 2018 Fifth International Conference on Emerging Applications of Information Technology (EAIT), Kolkata, India, 2018, pp. 1-4, doi: 10.1109/EAIT.2018.8470412.
- [13] H. P. Edmundson. 1969. New Methods in Automatic Extracting. J. ACM 16, 2 (April 1969), 264–285. <https://doi.org/10.1145/321510.321519>
- [14] H. P. Luhn, "The Automatic Creation of Literature Abstracts," in IBM Journal of Research and Development, vol. 2, no. 2, pp. 159-165, Apr. 1958, doi: 10.1147/rd.22.0159.

- [15] Dragomir R. Radev, Hongyan Jing, Małgorzata Styś, Daniel Tam, Centroid-based summarization of multiple documents, *Information Processing & Management*, Volume 40, Issue 6, 2004, Pages 919-938, ISSN 0306-4573, <https://doi.org/10.1016/j.ipm.2003.10.006>.
- [16] Dragomir R. Radev, Hongyan Jing, and Malgorzata Budzikowska. 2000. Centroid-based summarization of multiple documents: sentence extraction, utility-based evaluation, and user studies. In *NAACL-ANLP 2000 Workshop: Automatic Summarization*.
- [17] Goldstein-Stewart, Jade and Jaime G. Carbonell. "Summarization: (1) Using MMR for Diversity- Based Reranking and (2) Evaluating Summaries." NIST's TIPSTER Text Program (1998).
- [18] Marcu, D., & Gerber, L. (2001). An inquiry into the nature of multi document abstracts, extracts, and their evaluation. In *Proceedings of the 24th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval* (pp. 107-116).
- [19] Ji, P. (2006). Multi-document Summarization Based on Unsupervised Clustering. In: Ng, H.T., Leong, MK., Kan, MY., Ji, D. (eds) *Information Retrieval Technology. AIRS 2006. Lecture Notes in Computer Science*, vol 4182. Springer, Berlin, Heidelberg. [https://doi.org/10.1007/11880592\\_46](https://doi.org/10.1007/11880592_46)
- [20] Sarkar, Kamal. (2009). Sentence Clustering-based Summarization of Multiple Text Documents. *TECHNIA – International Journal of Computing Science and Communication Technologies*. 2. 325-335.
- [21] Abedinia, O., Amjady, N. and Ghasemi, A. (2016), A new metaheuristic algorithm based on shark smell optimization. *Complexity*, 21: 97-116. <https://doi.org/10.1002/cplx.21634>
- [22] A. Kogilavani and P. Balasubramanie, "Clustering based optimal summary generation using Genetic Algorithm," 2010 International Conference on Communication and Computational Intelligence (INCOCCI), Erode, India, 2010, pp. 324-329.
- [23] Rasmita Rautray, Rakesh Chandra Balabantaray, An evolutionary framework for multi document summarization using Cuckoo search approach: MDSCSA, *Applied Computing and Informatics*, Volume 14, Issue 2, 2018, Pages 134-144, ISSN 2210-8327, <https://doi.org/10.1016/j.aci.2017.05.003>.
- [24] Ganesan, Kavita A.. "ROUGE 2.0: Updated and Improved Measures for Evaluation of Summarization Tasks." *ArXiv abs/1803.01937* (2015): n. pag.
- [25] Ramina, M., Darnay, N., Ludbe, C., & Dhruv, A. (2020). Topic level summary generation using BERT induced Abstractive Summarization Model. 2020 4th International Conference on Intelligent Computing and Control Systems (ICICCS). doi:10.1109/iciccs48265.2020.9120997 10.1109/ICICCS48265.2020.9120997
- [26] Pradeepika Verma, Hari Om, MCRMR: Maximum coverage and relevancy with minimal redundancy based multi-document summarization, *Expert Systems with Applications*, Volume 120, 2019, Pages 43-56, ISSN 0957-4174, <https://doi.org/10.1016/j.eswa.2018.11.022>.
- [27] Minakshi Tomer, Manoj Kumar, Multi-document extractive text summarization based on firefly algorithm, *Journal of King Saud University - Computer and Information Sciences*, Volume 34, Issue 8, Part B, 2022, Pages 6057-6065, ISSN 1319-1578,

<https://doi.org/10.1016/j.jksuci.2021.04.004>.

(<https://www.sciencedirect.com/science/article/pii/S1319157821000902>)

- [28] Günes Erkan and Dragomir R. Radev. 2004. LexRank: graph-based lexical centrality as salience in text summarization. *J. Artif. Int. Res.* 22, 1 (July 2004), 457–479.
- [29] Rada Mihalcea and Paul Tarau. 2004. TextRank: Bringing Order into Text. In *Proceedings of the 2004 Conference on Empirical Methods in Natural Language Processing*, pages 404–411, Barcelona, Spain. Association for Computational Linguistics.
- [30] Rada Mihalcea and Paul Tarau. 2005. A Language Independent Algorithm for Single and Multiple Document Summarization. In *Companion Volume to the Proceedings of Conference including Posters/Demos and tutorial abstracts*.
- [31] Güneş, Erkan & Radev, Dragomir. (2004). LexPageRank: Prestige in Multi-Document Text Summarization.. 365-371.
- [32] Xiaojun Wan and Jianwu Yang. 2006. Improved Affinity Graph Based Multi-Document Summarization. In *Proceedings of the Human Language Technology Conference of the NAACL, Companion Volume: Short Papers*, pages 181–184, New York City, USA. Association for Computational Linguistics.
- [33] K. Sarkar, K. Saraf and A. Ghosh, "Improving graph based multidocument text summarization using an enhanced sentence similarity measure," 2015 IEEE 2nd International Conference on Recent Trends in Information Systems (ReTIS), Kolkata, India, 2015, pp. 359-365, doi: 10.1109/ReTIS.2015.7232905.
- [34] Raksha Agarwal, Niladri Chatterjee, Improvements in Multi-Document Abstractive Summarization using Multi Sentence Compression with Word Graph and Node Alignment, *Expert Systems with Applications*, Volume 190,2022,116154, ISSN 0957-4174, <https://doi.org/10.1016/j.eswa.2021.116154>.
- [35] Reimers, Nils & Gurevych, Iryna. (2019). Sentence-BERT: Sentence Embeddings using Siamese BERT-Networks. 3973-3983. 10.18653/v1/D19-1410.
- [36] Yang, XS. (2009). Firefly Algorithms for Multimodal Optimization. In: Watanabe, O., Zeugmann, T. (eds) *Stochastic Algorithms: Foundations and Applications*. SAGA 2009. Lecture Notes in Computer Science, vol 5792. Springer, Berlin, Heidelberg. [https://doi.org/10.1007/978-3-642-04944-6\\_14](https://doi.org/10.1007/978-3-642-04944-6_14)
- [37] Minakshi Tomer, Manoj Kumar, Multi-document extractive text summarization based on firefly algorithm, *Journal of King Saud University - Computer and Information Sciences*, Volume 34, Issue 8, Part B, 2022, Pages 6057-6065, ISSN 1319-1578, <https://doi.org/10.1016/j.jksuci.2021.04.004>.
- [38] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, Volume 1 (Long and Short Papers), pages 4171–4186, Minneapolis, Minnesota. Association for Computational Linguistics.


- [39] Nicole Peinelt, Dong Nguyen, and Maria Liakata. 2020. tBERT: Topic Models and BERT Joining Forces for Semantic Similarity Detection. In Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics, pages 7047–7055, Online. Association for Computational Linguistics.
- [40] Qazvinian, Vahed & Hassanabadi, Leila & Halavati, Ramin. (2008). Summarising text with a genetic algorithm-based sentence extraction. *International Journal of Knowledge Management Studies - Int J Knowl Manag Stud.* 2. 10.1504/IJKMS.2008.019750.
- [41] Kadhem, Suhad & Ali, Zuhair. (2018). Multi-Document Summarization using Fuzzy Logic and Firefly Algorithm.
- [42] H. Asgari, B. Masoumi and O. S. Sheijani, "Automatic text summarization based on multi-agent particle swarm optimization," 2014 Iranian Conference on Intelligent Systems (ICIS), Bam, Iran, 2014, pp. 1-5, doi: 10.1109/IranianCIS.2014.6802592.
- [43] Asma Al-Saleh and Mohamed El Bachir Menai. 2018. Ant Colony System for Multi-Document Summarization. In Proceedings of the 27th International Conference on Computational Linguistics, pages 734–744, Santa Fe, New Mexico, USA. Association for Computational Linguistics.

## **ACKNOWLEDGEMENT**

We are sincerely thankful to Dwarkadas J. Sanghvi College of Engineering for providing us with an opportunity to show our skills and to work on our project. We are extremely grateful to our project guide Prof.Pankaj Sonwane, who assisted us in conducting the research for this project and provided much-needed guidance through all stages of the process. In addition, we are extremely thankful to our respected Principal, Dr. Hari Vasudevan, and the Head of Department, Dr. Meera Narvekar, for giving us the support and guidance to work on this project. We also express our sincere gratitude to all the staff in the college who provided us with necessary resources and enriched our project with valuable suggestions. We express our sincere and heartfelt gratitude to all individuals without whose help it would not have been possible for us to complete the development within the prescribed time.



# PLAGIARISM



Similarity Report ID: oid:9832:35161082

PAPER NAME

BE Project Report.pdf

WORD COUNT

6789 Words

CHARACTER COUNT

38379 Characters

PAGE COUNT

30 Pages

FILE SIZE

876.2KB

SUBMISSION DATE

May 10, 2023 6:19 PM GMT+5:30

REPORT DATE

May 10, 2023 6:20 PM GMT+5:30

● 13% Overall Similarity

The combined total of all matches, including overlapping sources, for each database.

• 9% Internet database

• 8% Publications database

• Crossref database

• Crossref Posted Content database

• 7% Submitted Works database

● Excluded from Similarity Report

• Bibliographic material

• Quoted material

• Cited material

• Small Matches (Less than 10 words)

31