

Design & Development of a TLBO – based Metaheuristic Optimization Framework for Extractive Text Summarization

Nidhi A. Suthar¹, Heena K. Patel², Hiral Modi³, Rudrax Pandya⁴

Master's Degree in Computer Engineering, Sankalchand Patel University, Visnagar, India¹

Assistant. Professor, Sankalchand Patel College of Engineering, Visnagar, India²

Lecturer, Sankalchand Patel University, Visnagar, India³

Lecturer, Sankalchand Patel University, Visnagar, India⁴

nidhisthr611@gmail.com , hkpce_spce@spu.ac.in , hiralmodi50@gmail.com ,
rudrakshpandya2131@gmail.com

Abstract: Nowadays, various types of digital text data are available online and offline like news, research articles. The sudden explosion in digital text data has created a great need for efficient summarization methods that are capable of extracting important information quickly. It becomes increasingly difficult to extract any specific information from all this large data. The empirical basis of digital data has made automatic text summarization (ATS) an important area of research in natural language processing (NLP). Various methods are available for automatic text summarization like extractive text summarization and abstractive text summarization. Among various techniques, extractive summarization is widely used due to its simplicity and effectiveness. However, traditional heuristic and statistical approaches often struggle with redundancy, coherence and textual consistency. This research paper presents a comprehensive metaheuristic-based extractive text summarization using a Teaching-Learning based optimization algorithm. This algorithm optimizes sentence selection by balancing average, diversity and information. Benchmark datasets, including DUC2005 and DUC2006, along with evaluation metrics such as ROUGE and BLEU, are important for evaluating performance.

Keyword: Automatic Text Summarization, Metaheuristics, optimization, Extractive Summarization, NLP

I. INTRODUCTION

The fast growth of digital content on online platforms has caused a huge amount of text to be created every day. Communication channels such as email and instant messaging contribute significantly to this growth, with billions of messages exchanged each day, while social media platforms continue to generate vast amounts of short-form and multimedia-rich textual content. Although knowledge-centric platforms such as Wikipedia produce comparatively fewer contributions, they contain information of high semantic value. Collectively, these trends highlight the rapid accumulation of unstructured textual data in the digital ecosystem.

The increasing amount of text in many areas has made it harder for people to find, evaluate and understand the right information. Manual summarization of large document collections is both time-consuming and impractical, particularly given the scale and speed at which digital content is produced. Because of this, Automatic Text Summarization (ATS) has grown into a key area within Natural Language Processing (NLP). Its main aim is to create short summaries that are both clear and helpful, while still keeping the main ideas and information from the original text.

ATS techniques are usually divided into three main parts: extractive, abstractive and hybrid. Extractive methods focus on picking out the most relevant sentences from the original text. Abstractive methods, on the other hand, involve making new sentences that capture the main idea of the original content (1). Hybrid approaches combine the strengths of both methods to enhance summary quality. Among these approaches, extractive summarization remains widely adopted due to its simplicity, interpretability, and computational efficiency, particularly in large-scale and multi-document scenarios (2).

Despite their effectiveness, traditional extractive summarization methods based on heuristics or statistical features often suffer from limitations such as redundancy, lack of coherence, and insufficient contextual coverage (3). To overcome these challenges, optimization-based approaches—especially metaheuristic algorithms—have gained increasing attention in recent years. These methods formulate sentence selection as an optimization problem, enabling the generation of summaries that balance relevance, diversity, and coverage more effectively than conventional techniques. This document provides a detailed look at extractive multi-document text summarization that use metaheuristics and highlights how these approaches can address the shortcomings of traditional techniques.

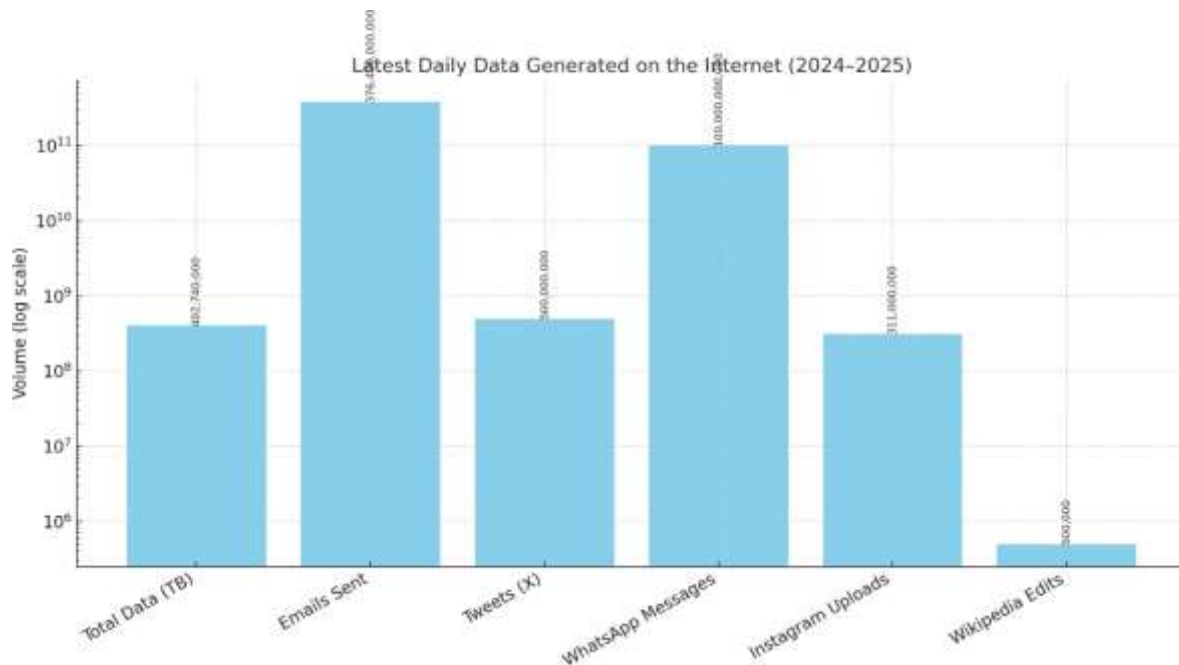


Figure 1. Volume of data created worldwide from 2024 to 2025 [1]

The remainder of this paper is organized as follows. Section 2 presents the classification of Automatic Text Summarization techniques. Section 3 discusses extractive text summarization and existing summarization approaches. Section 4 reviews metaheuristic optimization algorithms applied to extractive summarization. Section 5 describes commonly used evaluation metrics. Finally, Section 6 concludes the paper and highlights future research directions

II. CLASSIFICATION

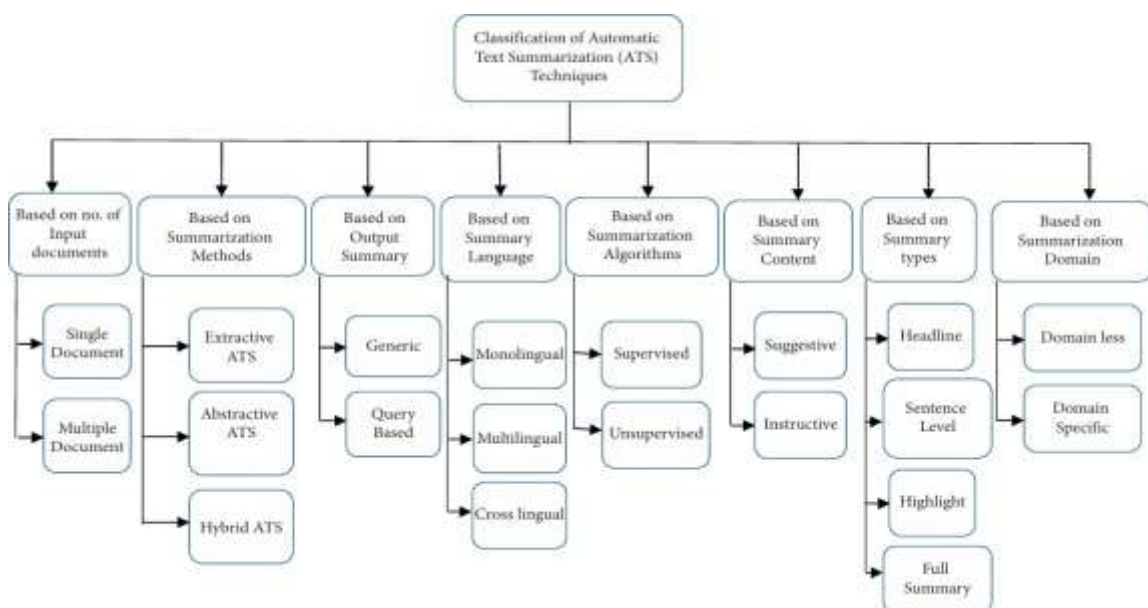


Figure 2. Classification of Automatic Text Summarization [2]

Classification Basis	Categories	Description
Based on Input Size	Single-document, Multi-document	Single-document summarization generates a summary from one document, whereas multi-document summarization produces a summary from a collection of related documents [4].
Based on Summarization Approach	Extractive, Abstractive	Extractive summarization selects important sentences directly from the source text, while abstractive summarization generates new sentences that may not explicitly appear in the original document [4].
Based on Output Summary	Generic, Domain-specific, Query-based	Generic summaries treat all text equally; domain-specific summaries use domain knowledge for improved accuracy; query-based summaries generate content relevant to a user-defined query [5].
Based on Summary Language	Monolingual, Multilingual, Cross-lingual	Monolingual systems produce summaries in the same language as the input; multilingual systems handle multiple languages; cross-lingual systems generate summaries in a different language from the input [6].
Based on Summarization Algorithm	Supervised, Unsupervised	Supervised methods require labeled training data, whereas unsupervised methods operate without human-annotated datasets [6].
Based on Summary Content	Indicative, Informative	Indicative summaries provide a general overview of the document, while informative summaries include all key points from the original text [7].
Based on Summary Type	Headline, Sentence-level, Highlight, Full summary	Headline summaries are very short; sentence-level summaries produce a single sentence; highlight summaries use bullet points; full summaries depend on the required compression ratio [8], [9].
Based on Summarization Domain	Domain-agnostic, Domain-specific	Domain-agnostic approaches work across multiple domains, whereas domain-specific approaches are tailored to texts from a particular domain [3].

III. EXTRACTIVE TEXT SUMMARIZATION

An automatic extractive text summarization method works by selecting and combining the most important sentences from a document or set of documents [3]. Unlike abstractive summarization, this approach is simpler and requires less computer power. Since the sentences used in the summary come directly from the original text, the summary keeps the original wording, which helps maintain accuracy and reduces the risk of changing the meaning [10]. The process of extractive text summarization usually has three main steps: pre-processing, processing and post-processing as shown in Figure 3 [3].

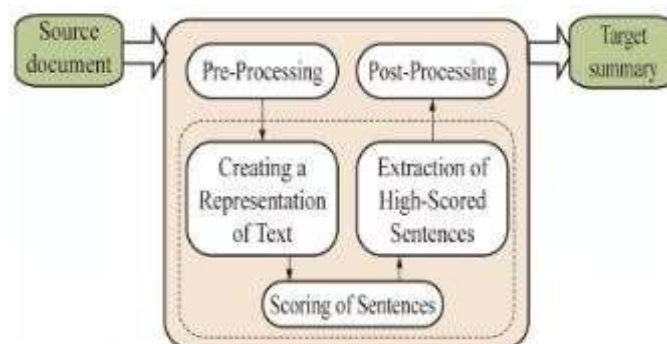


Figure 3: The architecture of an extractive text summarization [4]

Pre-Processing: Pre-processing aims to convert the original text into a structured representation. This stage typically includes: a) Sentence boundary detection, where punctuation marks such as full stops are used to identify

sentence endings in English text. b) Stop-word elimination, which removes commonly occurring words that carry little semantic meaning and do not contribute useful information to the summarization task. c) Stemming, which reduces words to their root or base form in order to emphasize their underlying meaning [11].

Processing: The processing stage identifies and computes the features that influence sentence importance. These features are then assigned appropriate weights using a weight-learning approach. Based on the feature-weight formulation, a score is calculated for each sentence. The sentences that get the highest scores are included in the final summary [11].

Post-processing: Post-processing involves organizing the selected sentences into a logical and coherent order. This stage also addresses linguistic issues such as redundancy and anaphora resolution to improve the overall readability and coherence of the generated summary.

Challenges: The next points list several difficulties connected with extractive summarization [11]:

The extracted sentences are longer compared to the normal length. As a result, the segments' irrelevant portions are also extracted, taking up space.

Extractive summaries do not capture important or useful information that is often dispersed throughout sentences (barring the summary being long enough to encompass all those sentences).

There can be errors in the presentation of conflicting information.

IV. EXISTING TEXT SUMMARIZATION APPROACHES

In this subsection, a brief description of existing text summarization methods is provided. These methods can be used on their own or combined to make extractive text summarization systems work better. The decision on which technique to use depends on factors like how long the document is, how much needs to be summarized and how complicated the sentences are.

Statistical-Based Methods: Statistical methods use analysis of various features to find out which words and phrases are most important in the original text. In these approaches, terms that are more frequent, appear in advantageous positions, or exhibit higher statistical importance are considered more significant for summarization purposes [11].

Graph-Based Method: Graph-based approaches are unsupervised techniques that rank keywords, sentences, or phrases using graph representations. In this method, documents or sentences are represented as nodes, and edges connect pairs of nodes that share common information or semantic similarity. The importance of each node is determined based on its position and connectivity within the graph structure [12].

Concept-Based Method: Concept-based extractive text summarization focuses on identifying important concepts or ideas present within a document. These concepts are then used to select representative sentences that best convey the main information of the text [3].

Topic-Based Method: The objective of topic-based extractive text summarization is to reduce document length by identifying and emphasizing the major topics or themes present in the text. Sentences associated with dominant topics are selected to form the summary [3].

Semantic-Based Method: Semantic-based approaches consider the semantic relationships between words when computing document or sentence similarity. These methods focus on word meanings and the implicit semantic connections that exist between words, and consequently between sentences and documents, rather than relying solely on surface-level statistical features [13].

Sentence Centrality Based Method: Sentence centrality-based methods identify key and critical sentences within a document or document cluster by determining the importance of words and sentences relative to the overall content. A common approach involves finding the center point of a group of similar documents in a vector space and then locating the sentences that are closest to this center point [3].

Machine-Learning Based Method: Machine learning approaches view text summarization as a supervised classification task for each sentence. A trained model sorts sentences into two groups: those that are included in the summary and those that are not. This is done by using features that the model has learned from previously labeled training data[3].

Deep-Learning Based Method: Methods that use deep learning rely on complex neural structures to represent sentences and create summaries. Models such as Long Short-Term Memory networks (LSTMs), Convolutional Neural Networks (CNNs), Recurrent Neural Networks (RNNs) and Transformer-based models like BERT and GPT are commonly used for extractive summarization tasks.

Optimization based: Optimization-driven methods treat multi-document text summarization as a problem that involves balancing several goals at the same time, such as relevance, coverage, redundancy and coherence. Various optimization techniques have been proposed to effectively address these objectives in extractive summarization systems [14].

V. LITERATURE

Text summarizing was first attempted in the 1960s and 1970s, mostly with rule-based systems that choose sentences according to pre-established standards such sentence length, keyword frequency, and document position. Scholars started investigating statistical methods for text summarization in the 1990s. In the text summarization, supervised learning approaches became popular in the 2000s and 2010s. Researchers used machine learning methods such as decision trees and support vector machines to determine which sentences are important or not for creating summaries. With the help of more advanced models like recurrent neural networks, transformer-based systems like BERT and GPT, and convolutional neural networks, it's now possible to build smarter and more context-aware summarization tools. late 1990s and early 2000s, the graph-based text summarization methods took center stage. Techniques such as TextRank, inspired by Google PageRank, modeled documents as phrase graphs with semantic similarity edges. The concepts of TextRank were expanded to document summarization with the introduction of LexRank by Erkan and Radev in 2004. It achieved better summarization performance by measuring the semantic relatedness and significance of sentence vectors using cosine similarity.

Table 2: Comparison of extractive text summarization-based methods

Sr. No.	Title / Study	Technique	Approach	Challenges Addressed	Future Scope
1.	Extractive Text Summarization using Metaheuristic Approach [15]	Genetic Algorithm (GA),	ROUGE-1 = 0.3843, ROUGE-2 = 0.2584.	Processing of large & complex data sets (news corpus, Indian languages), Enhancing precision and recall in the summaries	Scale to more low-resource languages, Real-time applications of summarization for news and social media
2.	Metaheuristic optimization Using Sentence Level Semantics for Extractive Document Summarization [16]	Genetic Algorithm (GA),	ROUGE-1 $\approx$ 0.55, ROUGE-2 $\approx$ 0.27	Redundancy in summaries, Balancing relevance and diversity, Preserving semantic meaning other than through statistics	Incorporate with neural embeddings (word2vec, BERT) for more semantically-rich features, Experiment on multilingual datasets

3.	Summarization of scholarly articles using BERT AND BiGRU: deep learning based extractive approach [17]	BERT and BiGRU	ArXiv: ROUGE1 = 46.7, ROUGE2 = 19.4, ROUGE-L = 35.4 Pubmed: ROUGE1 = 47.0, ROUGE2 = 21.3, ROUGE-L = 39.7	BERT for the extraction of information from lengthy academic texts. The suggested method involved incorporating BiGRU on top of BERT to more effectively capture the global	For long sequences, the learning of GRU is very difficult
				context of the papers as well as assimilating BERT to handle lengthy texts.	
4.	Greedy optimization Method for Extractive Summarization of Scientific Articles [18]	Greedy algorithm along with VNS	ArXiv: ROUGE1 = 0.43, ROUGE2 = 0.12 Pubmed: ROUGE1 = 0.40, ROUGE2 = 0.13	Two methods were compared to achieve the best ROUGE score: VNS technique and a Greedy algorithm. Despite similar performance, the Greedy approach demonstrated faster completion times.	Greedy algorithms are not suitable for solving complex optimization problems
5.	A Hybrid Text Summarization Approach Using Neural Networks and Metaheuristic Algorithms [19]	Genetic Algorithm (GA) and Particle Swarm Optimization (PSO)	ROUGE-1 = 0.612, ROUGE-2 = 0.417, and ROUGE-L = 0.398.	Pre-processing, Generation of Summary, Feature Extraction	Real-time Summarization, Scalability
6.	Extractive single document summarization using multi-objective modified cat swarm optimization approach: ESDS-MCSO [20]	cat swarm optimization	ROUGE-1 = 0.5324, ROUGE-2 = 0.3126	Redundancy, Coherence, Coverage Balance	Multi-document Summarization, Real-time Applications
7.	Text Summarization Using Swarm-Based and Genetic-Based Methods in Natural Language Processing: A Comparative Review [21]	Ant Colony Optimization (ACO) and Particle Swarm Optimization (PSO)	ROUGE-1 score of 0.58	Redundancy, Coherence, Evaluation Metrics	Multi-document Summarization, Real-time Summarization

The diversity of approaches, tactics, and challenges discussed in the current study is reflected in the comparative table of extractive text summarizing methodologies. Genetic Algorithm (GA), BERT-based hybrid models, greedy optimization, and swarm-based approaches such as Particle Swarm Optimization (PSO) and Ant Colony Optimization (ACO) were some of the various algorithms that were evaluated. RoUGE-1 and RoUGE-2 scores for genetic algorithms, which are often employed to optimize summary quality and feature selection, were between 0.43 and 0.52 and 0.25 and 0.27, respectively. This reflects consistent but moderate performance for texts that are domain-specific or complex. BERT and hybrid models, in contrast, delivered stronger results for scholarly article summarization, with ROUGE-1 scores reaching 0.61 and ROUGE-L up to 39.7, particularly excelling in maintaining cohesion and coverage with large, multi-topic papers.

Redundancy and coherence are managed efficiently by swarm-based methods such as PSO and ACO, which according to reports attain ROUGE-1 scores of around 0.58 and are keen on evaluation metrics constructed for

multi-document summarizing purposes. Greedy optimization methods, especially those augmented by variable neighborhood search (VNS), also performed competitively, though their scalability for highly complex optimization scenarios remains an area for future work. Across all approaches, the table highlights ongoing challenges, including processing lengthy or multilingual documents, minimizing redundancy, presenting nuanced information, and adapting models for broader domains or real-time applications. Researchers suggest that advancing scalability and integration with sophisticated models like BERT, as well as developing efficient solutions for hybrid and multi-document settings, will be pivotal for future growth in this field.

VI. PROPOSED OPTIMIZATION ALGORITHM

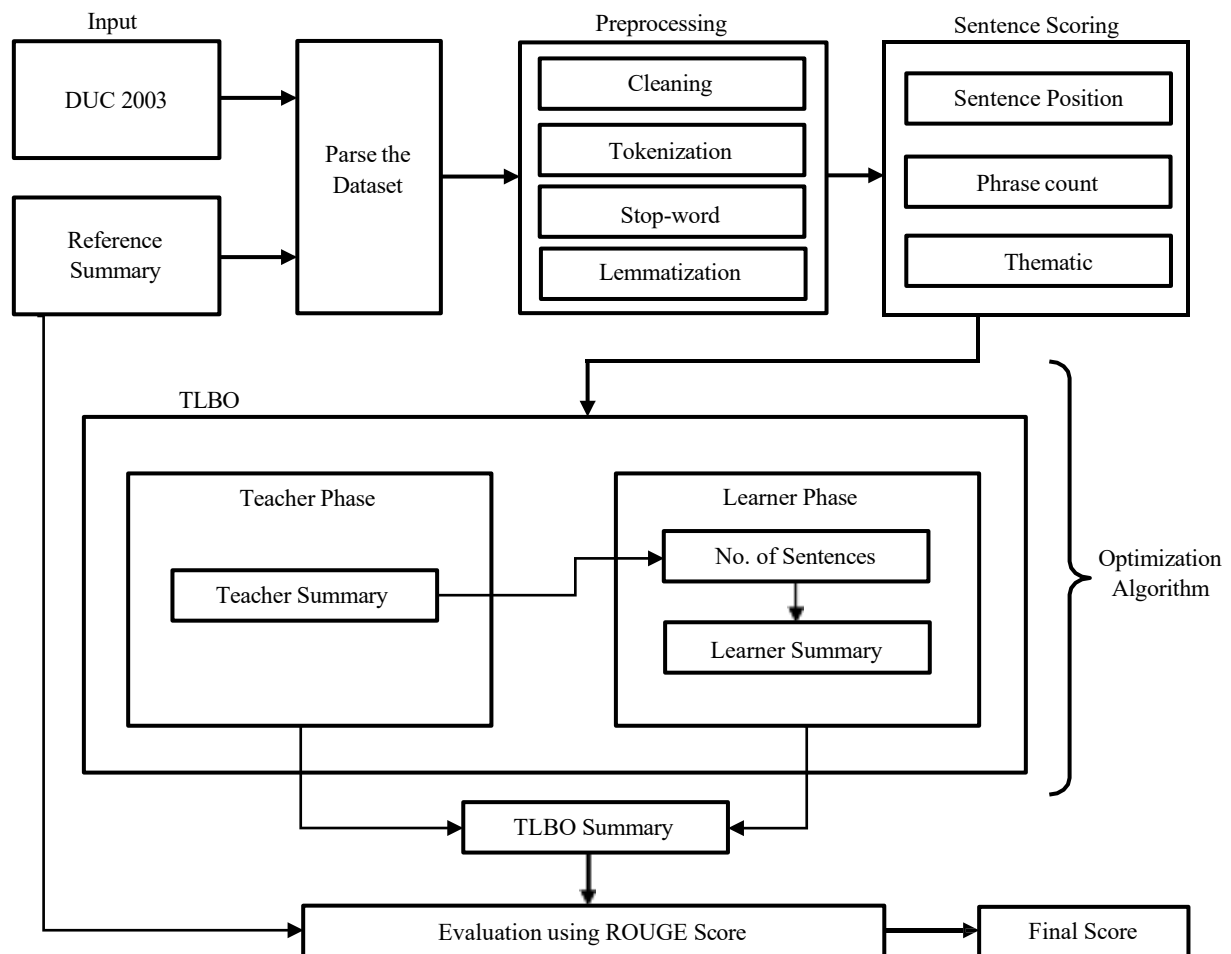


Figure 4: Proposed Optimization Algorithm

6.1 Dataset: The DUC 2003 dataset is benchmark collection made by the National Institute of Standards and Technology (NIST) for research in multi-document text summarization. It is designed for query-focused extractive summarization, which means it helps create a short summary of multiple documents based on a specific question or topic. The dataset includes 90 groups of documents, called topics and each group has about 20 to 25 news articles, making a total of 1923 documents. For each topic, there are human-written summaries that are usually around 100 words long. These summaries help researchers test and compare the performance of different summarization systems reliably.

All the documents in DUC 2003 are written in English, making it suitable for real-world summarization studies. The dataset also allows for the extraction of various features commonly used in summarization models, such as sentence-level statistical data, term frequency (TF) and TF-IDF features, features based on the position of sentences, and features based on similarity between sentences. Because of its organized structure, clear rules about length, and the availability of reference summaries, DUC 2003 is an important resource for evaluating and comparing extractive multi-document summarization methods.

6.2 Parse the Dataset: Parsing the DUC 2003 dataset needs a clear method to get the documents, topics and summaries from its organized structure. It's set up with different topic folders, and each folder has news articles along with summaries written by people. The first step is to look through the folders, find the topic numbers, and get each document inside the group.

Each document has tags like the document ID, headline and the main text. When you parse the data, you take out the tags the real text. You also get the summaries and link them to the right topic number so you can use them for checking how well the system works. After you extract everything, you put the clean text into a format like common words, finding important features, and making summaries. Good parsing makes sure the documents and summaries match up properly, which is important for training and testing multi-document summarization systems.

6.3 Preprocessing: The preprocessing stage is an important part of Natural Language Processing (NLP) that prepares raw text data for further analysis and modeling. Text from real-world sources often has problems like punctuation mistakes, unusual symbols, uneven formatting and extra words, which can make it harder for machine learning or summarization tools to work effectively. Because of this, the main goal of preprocessing is to clean up, organize and present the text in a suitable way for extracting features and training models.

Common steps in preparing text data include cleaning the text, breaking it into smaller parts, removing common but not very meaningful words, and simplifying words to their basic forms. Cleaning the text means taking out extra symbols and fixing any formatting problems. Breaking the text into smaller parts is called tokenization, and it splits the text into words or sentences. Removing common words like "the", "is" and "and" is known as eliminating stop words. Simplifying words to their root form can be done through stemming or lemmatization. These steps help make the text easier to work with, reduce complexity and improve the performance of text-based systems like classification, sentiment analysis and summarization tools.

6.3.1 Text Cleaning: Text cleaning is the first and most important step in preparing text for multi-document summarization. In real-world datasets, like news articles used in challenges such as DUC, documents often contain things like punctuation, special symbols, HTML or XML tags, problems by making it harder to extract useful features, score sentences, and compare how similar documents are. So, the goal of text cleaning is to remove these unwanted understand and work with the actual important information.

In multi-document summarization, this step is especially important because it involves combining several related documents into one short summary. For example,

We have original sentence like this –

S1: The Prime Minister visited New Delhi on Monday!!!

S2: In New Delhi, the Prime Minister announced new economic reforms.

S3: According to officials, the reforms will impact 5000 small businesses across the country.

S4: Experts said the announcement was 'a major step forward' for economic growth.

After cleaning –

S1: The Prime Minister visited New Delhi on Monday

S2: In New Delhi the Prime Minister announced new economic reforms

S3: According to officials the reforms will impact small businesses across the country

S4: Experts said the announcement was a major step forward for economic growth

6.3.2 Tokenization: Tokenization is an important step in preparing text for multi-document summarization. It breaks down the cleaned text into smaller pieces called tokens, which can be individual words or whole sentences. In multi-document summarization, this process helps organize large volumes of text from different sources into smaller, easier-to-handle sections. These tokens are then used for tasks like removing common words, extracting important features, measuring how similar sentences are, and determining which sentences are most important.

For example,

We have sentences after text cleaning like this –

S1: The Prime Minister visited New Delhi on Monday

S2: In New Delhi the Prime Minister announced new economic reforms

S3: According to officials the reforms will impact small businesses across the country

S4: Experts said the announcement was a major step forward for economic growth

After tokenization –

S1: [The, Prime, Minister, visited, New, Delhi, on, Monday]

S2: [In, New, Delhi, the, Prime, Minister, announced, new, economic, reforms]

S3: [According, to, officials, the, reforms, will, impact, small, businesses, across, the, country]

S4: [Experts, said, the, announcement, was, a , major, step, forward, for, economic, growth]

6.3.3 Stop-word removal: The process of removing stop words is an important step in preparing documents for multi-document summarization. Stop words are common words like 'the', 'is', 'in', 'on', 'to', 'for' and 'a' that do not add much value when analyzing content. In multi-document summarization, large groups of documents often contain many of these frequently used but not very informative words. By removing them, we can reduce unnecessary information, simplify the data, and improve the effectiveness of methods like TF-IDF and similarity checks.

For example,

We have sentences after tokenization like this:

S1: [The, Prime, Minister, visited, New, Delhi, on, Monday]

S2: [In, New, Delhi, the, Prime, Minister, announced, new, economic, reforms]

S3: [According, to, officials, the, reforms, will, impact, small, businesses, across, the, country]

S4: [Experts, said, the, announcement, was, a , major, step, forward, for, economic, growth]

After removing stop-words,

S1: [Prime, Minister, visited, New, Delhi, Monday]

S2: [New, Delhi, Prime, Minister, announced, new, economic, reforms]

S3: [According, officials, reforms, impact, small, businesses, across, country]

S4: [Experts, said, announcement, major, step, forward, economic, growth]

6.3.4 Lemmatization: Lemmatization is a technique used in Natural Language Processing to change words into their base or dictionary form, called the lemma, and it considers how words are used grammatically. Unlike stemming, which just cuts off the ending of words, lemmatization gives more accurate root forms based on language rules. In multi-document summarization, it helps group different forms of a word, like announced, announcing, and announces, into one common form (announce), which improves how often words are counted, how similar words are judged, and the overall quality of the summary.

For example,

We have sentences after removing stop-words like this:

S1: [Prime, Minister, visited, New, Delhi, Monday]

S2: [New, Delhi, Prime, Minister, announced, new, economic, reforms]

S3: [According, officials, reforms, impact, small, businesses, across, country]

S4: [Experts, said, announcement, major, step, forward, economic, growth]

After lemmatization,

S1: [Prime, Minister, visit, New, Delhi, Monday]

S2: [New, Delhi, Prime, Minister, announce, new, economic, reform]

S3: [According, official, reform, impact, small, business, across, country]

S4: [Expert, say, announcement, major, step, forward, economic, growth]

6.4 Sentence Scoring: Sentence evaluation is a key part of extractive multi-document summarization, where each sentence gets a number that shows important it is. The main goal of this process is to find and pick the most useful and relevant sentences from different documents to make a clear and concise summary. Instead of rewriting the content, extractive systems look at sentences based on certain factors and choose the ones with the highest scores.

Sentence scores are usually based on several factors like term frequency (TF _IDF), where words that appear more often are considered more important, the position of the sentence in the text, how similar it is to other sentences, whether it contains key words, and how the main ideas are spread out.

In multi-document summarization, this scoring helps reduce repeated information and makes sure the selected sentences cover the main points from all the documents. By giving appropriate weight to each feature, sentence evaluation plays a big role in creating accurate, logical and meaningful summaries.

6.4.1 Sentence Position: The location of a sentence plays a major role in how it is scored during extractive summarization. This is based on the idea that where a sentence appears in a document often shows how important it is. In news stories and formal writing, the first sentences usually contain the most important details, like the main event, a key announcement, or a brief summary of what the text about. Because of this, sentences that come early in a document-especially the first ones, and sometimes the last-tend to get higher scores.

A commonly used normalization formula is:

$$\text{Sentence Position Score (S}_{\text{pos}}) = 1 - \frac{i-1}{N}$$

Where,

i = position of the sentence in the document

N = total number of sentences in the document

For example,

We have original sentence like this –

S1: Prime Minister visit New Delhi Monday

S2: New Delhi Prime Minister announce new economic reform

S3: According official reform impact small business across country

S4: Expert say announcement major step forward economic growth

Now calculate position scores:

$$\text{For S1 (i=1): } S_{\text{pos}} = 1 - \frac{1-1}{4} = \frac{4}{4} = 1.0$$

$$\text{For S2 (i=2): } S_{\text{pos}} = 1 - \frac{2-1}{4} = \frac{3}{4} = 0.75$$

$$\text{For S3 (i=3): } S_{\text{pos}} = 1 - \frac{3-1}{4} = \frac{2}{4} = 0.50$$

$$\text{For S4 (i=4): } S_{\text{pos}} = 1 - \frac{4-1}{4} = \frac{1}{4} = 0.25$$

6.4.2 Phrase count: The number of phrases is an important factor used to evaluate sentences by checking how many meaningful or common phrases are present. A phrase consists of two or more words that together have a specific meaning, like economic reform or New Delhi. In the case of summarizing multiple documents, sentences with more important phrases are documents. So, counting these phrases helps identify sentences that have more content, which is helpful for making summaries.

A commonly used normalization formula is:

$$S_{\text{phrase}} = \frac{P_s}{P_{\text{max}}}$$

Where,

P_s = Number of important phrases in a sentence

P_{max} = Maximum phrase count among all sentences

For example,

We have original sentence like this –

S1: Prime Minister visit New Delhi Monday

S2: New Delhi Prime Minister announce new economic reform

S3: According official reform impact small business across country

S4: Expert say announcement major step forward economic growth

Assume important phrases identified across documents are:

Prime Minister, New Delhi, economic reform, small business, economic growth

Now count phrases in each sentence:

S1: (Prime Minister, New Delhi) = 2 phrases

S2: (New Delhi, Prime Minister, economic reform) = 3 phrases

S3: (small business) = 1 phrases

S4: (economic growth) = 1 phrases

Here, P_{max} = 3

Now compute normalized score:

S1: 2/3 = 0.67

S2: 3/3 = 1.00

S3: 1/3 = 0.33

$$S4: 1/3 = 0.33$$

6.4.3 Thematic: The idea behind sentence scoring based on theme is that words that show up often in many documents are likely to be the main ideas or topics those documents are about. These common words are called thematic words. When summarizing multiple documents, sentences that have more of these thematic words are considered more important because they better capture the main subject of all the documents together. So, counting how many thematic words are in each sentence helps find the best sentences to include in the summary.

A commonly used normalization formula is:

$$S_{\text{thematic}} = \frac{T_s}{T_{\text{max}}}$$

Where,

T_s = Number of thematic words in a sentence

T_{max} = Maximum thematic word count among all sentences

For example,

We have original sentence like this –

S1: Prime Minister visit New Delhi Monday

S2: New Delhi Prime Minister announce new economic reform

S3: According official reform impact small business across country

S4: Expert say announcement major step forward economic growth

Assume important phrases identified across documents are:

Prime, Minister, New, Delhi, reform, economic

Now count thematic words in each sentence:

S1: (Prime, Minister, New, Delhi) = 4

S2: (New, Delhi, Prime, Minister, economic, reform) = 6

S3: (reform) = 1

S4: (economic) = 1

Here, $T_{\text{max}} = 4$

Now calculate thematic scores:

S1: $4/6 = 0.67$

S2: $6/6 = 1.00$

S3: $1/6 = 0.17$

S4: $1/6 = 0.17$

6.5 TLBO: Teaching-Learning-Based Optimization (TLBO) is an optimization method developed in 2011 by Ravindra V. Rao. It is based on how teaching and learning work in a classroom, where a teacher helps students improve their knowledge. In this approach, each possible solution is treated like a student, and the best solution in the group acts as the teacher. The goal of the algorithm is to improve the quality of solutions by sharing and building knowledge among all members of the group.

TLBO works in two main steps: the Teacher Phase and the Learner Phase. In the Teacher Phase, the teacher tries to guide the group towards better solutions by adjusting the average of all solutions. In the Learner Phase, the students learn from each other to make more improvements. These steps are repeated over until a stopping condition, like reaching a set number of iterations, is met. Because it has a simple structure and doesn't require many special settings, TLBO is easy to use and works well for many different types of optimization problems.

Teacher phase: It is a mark of an effective teacher to elevate their students' knowledge level to their own level. However, in actuality, this is not feasible, and a teacher's capacity to raise the class mean is limited by the students' abilities [27].

Here, Eq. (9) is employed to reweight the text features' weights with the mean value of the text features and the given teacher solution.

The most frequent sentence in the documents is chosen to be a teacher, and they subsequently work hard to raise the class's general level.

$$X_{\text{new}} = X_{\text{old}} + \text{rand} (X_{\text{teacher}} - \text{TF} * X_{\text{mean}}) \dots \dots \dots \text{eq}(1)$$

where X_{new} and X_{old} stands for the candidate solutions both before and after updating, or the individual's positions before and after learning. The position of the teacher, who is the best member of the population, is represented by $X_{teacher}$. The mean is the average search agents' level of the population. The mean is the average level of the population of search agents. "rand" is a random number between 0 and 1, and "TF" is a teaching factor that regulates the variation in the mean value. TF, a heuristic step chosen randomly with an equal probability, is either 2 or 1, where

$$TF = \text{round}(1 + \text{rand}(0,1) \{2-1\}) \dots \dots \dots \text{eq}(2)$$

Learner phase: Along with learning new information from the instructor, students can increase their knowledge by interacting with others. A student can arbitrarily take up information from a higher-graded other people throughout their mutual learning process. Take into consideration that the randomly chosen answer for each solution X_q is X_q .

After evaluating both of the solutions, X_q and $X_{q'}$, if $X_q < X_{q'}$, then X_q is modified in the manner described below:

$$\text{New_}X_{q,\text{dim}} = X_{q,\text{dim}} + r_z(X_{q,\text{dim}} - X_{q',\text{dim}}) \dots \dots \dots \text{eq}(3)$$

New X_q stands for the new solution for q^{th} . Otherwise, update the X_q :

$$\text{New_}X_{q,\text{dim}} = X_{q,\text{dim}} + r_z(X_{q',\text{dim}} - X_{q,\text{dim}}) \dots \dots \dots \text{eq}(4)$$

The X_q will not be updated if $X_q = X_{q'}$

Algorithm:

Step – 1: Process the document

- Split Document D into S sentences
- Extract sentence features

Step – 2: Initialize population X_i ($i = 1$ to N)

- Each X_i is a binary vector of length S
- Ensure summary length constraint L is satisfied

Step – 3: Evaluate fitness $f(X_i)$ for each student

Step – 4: Set iteration = 1

Step – 5: While iteration $\leq$ MaxIter

Teaching Phase -----

Identify the best solution as Teacher ($X_{teacher}$)

Compute mean of population(M)

For each student X_i

Generate Teaching Factor

$TF = \text{round}(1 + \text{rand}(0,1))$

Generate random number $r \in (0,1)$

$X_{new} = X_i + r * (X_{teacher} - TF * X_{mean})$

Convert X_{new} to binary using sigmoid/threshold

Repair X_{new} to satisfy length constraint L

If $f(X_{new})$ better than $f(X_i)$

$X_i = X_{new}$

End If

End For

Learner Phase -----

For each student X_i

Randomly select another student X_j ($i \neq j$)

Generate random number $r \in (0,1)$

If $f(X_i)$ better than $f(X_j)$

$$X_{\text{new}} = X_j + r * (X_i - X_j)$$

Else

$$X_{\text{new}} = X_j + r * (X_j - X_i)$$

End If

Convert X_{new} to binary

Repair X_{new} to satisfy length constraint L

If $f(X_{\text{new}})$ better than worse solution

Replace worse solution with X_{new}

End If

End For

Update iteration = iteration + 1

End While

Step – 6: Return best solution found (X_{best})

VII. Evolution Metrics

The quality of the summaries created is often checked using different evaluation measures. These measures help determine how effectively the summary captures the main points from the original text or documents.

7.1 ROUGE: ROUGE stands for Recall-Oriented Understudy for Gisting Evaluation. It is a way to check how well machine-made summaries match human-made summaries [28]. Here are some key ROUGE metrics:

7.1.1 ROUGE-1: The unigram, also called ROUGE-1, is used to measure how different a candidate summary is from a reference summary [29]. The recall of n-grams between a reference summary and a candidate summary is referred to ROUGE-N.

Precision (P): As a proportion of all generated summary unigrams, Precision indicates how many overlapping unigrams between a generated summary and a reference summary exist. You can calculate it with the given formula:

$$P = \text{Number of overlapping unigrams} / \text{Total no. of unigrams in the generated summary} \dots \text{eq(11)}$$

Recall(R): The recall is found by checking how many of the single words (unigrams) from the reference summary are also in the generated summary, compared to the total number of single words in the reference summary. This is done using this formula:

$$R = \text{Number of overlapping unigrams} / \text{Total number of unigrams in the reference summary} \dots \text{eq(12)}$$

F1-Score: Precision and recall are combined into one overall measure called the F1 score, which is the harmonic mean of the two. The formula used to calculate it is as follows:

$$\text{F1-Score} = (2 * \text{Precision} * \text{Recall}) / (\text{Precision} + \text{Recall}) \dots \text{eq(13)}$$

Example: The overlap in unigrams between the reference summary and the generated summary is computed by ROUGE-1.

Reference Summary: "Global warming is a pressing issue affecting our planet."

Generated Summary: "The issue of global warming is a concern for our planet."

Overlap Calculation: ROUGE-1 is employed to estimate the overlap of unigrams between the reference summary and the computer summary.

Number of overlapping Unigrams = 5

Total number of unigrams in the generated summary = 11

Total number of unigrams in the reference summary = 9

i. Precision(P):

$Precision(P) = \text{Number of overlapping unigrams} / \text{Total no. of unigrams in the generated summary}$

$Precision(P) = 5/11 \approx 0.4545$

ii. Recall(R):

$Recall(R) = \text{Number of overlapping unigrams} / \text{Total number of unigrams in the reference summary}$

$Recall(R) = 5/9 \approx 0.5556$

iii. F1- score:

$F1-Measure = (2 * Precision * Recall) / (Precision + Recall)$

$F1-Measure = (2 * 0.4545 * 0.5556) / (0.4545 + 0.5556) \approx 0.5000$

ROUGE-2 measures the overlap of bigrams, which are pairs of words, while ROUGE-3 checks trigrams, which are three-word sequences, and ROUGE-4 looks at 4-grams, which are four-word sequences. ROUGE-n is used to evaluate the quality of automatically generated summaries or translations by calculating how much n-gram overlap there is between the generated text and the reference text. N-grams are simply sequences of n words.

7.1.2 ROUGE-L: ROUGE-L (R-L) measures how similar two summaries are by looking at the longest sequence of words that appear in both the reference summary and the candidate summary.

Example: A simple example is as follows. ROUGE-L: Let's look at a reference summary and a generated summary from a study about climate change.

Reference Summary: "Global warming is a pressing issue affecting our planet."

Generated Summary: "The issue of global warming is a concern for our plane."

The longest common subsequence is: "global warming is a planet"

Length of the longest common subsequence: 6

Word count of the generated summary = 11

Word count of the reference summary = 9

i. Precision(P):

$Precision(P) = \text{Length of the longest common subsequence} / \text{Total no. of words in the generated summary}$

$Precision(P) = 6 / 11 \approx 0.5455$

ii. Recall(R):

$Recall(R) = \text{Number of overlapping unigrams} / \text{Total no. of words in the reference summary}$

$Recall(R) = 6 / 9 \approx 0.6667$

iii. F1-score:

$F1-Measure = (2 * Precision * Recall) / (Precision + Recall)$

$F1-Measure = (2 * 0.5455 * 0.6667) / (0.5455 + 0.6667) \approx 0.6000$

7.1.3 ROUGE-S*: ROUGE-S* (R-S*) measures the ratio of overlapping skip-bigrams between a reference summary and a candidate summary.

Example: Here's a simple example. ROUGE-S*: Let's think about the situation where we have both a reference summary and a summary that has been generated.

Reference Summary: "The cat sat on the mat."

Generated Summary: "The dog sat on the rug."

Assume for the sake of simplicity that skip-bigrams with a maximum skip length of 1 are under consideration.

Total number of skip-bigrams in the reference summary = 4 (The, sat), (cat, on), (sat, the), (on, mat)

Total number of skip-bigrams in the generated summary = 5 (The, sat), (dog, sat), (sat, on), (on, the), (the, rug)

Number of overlapping skip-bigrams = 3 (The, sat), (sat, on), (on, the)

i. Precision(P):

$Precision(P) = \text{Number of Overlapping skip bigrams} / \text{Total no. of skip bigrams in the generated summary}$

$Precision(P) = 3 / 5 = 0.6$

ii. Recall(R):

Recall (R) = Number of Overlapping skip bigrams / Total number of skip bigrams in the reference summary

$$\text{Recall}(R) = 3 / 4 = 0.75$$

iii. F1-score:

$$F1 - \text{Measure} = (2 * \text{Precision} * \text{Recall}) / (\text{Precision} + \text{Recall})$$

$$F1 - \text{Measure} = (2 * 0.6 * 0.75) / (0.6 + 0.75) \approx 0.6667$$

7.1.4 ROUGE-SU*: By employing skip bigrams and the unigram as a counting unit, it expands upon ROUGE-S*. The number of skip words is indicated by the "*". For example, ROUGE-SU4 defines that bi-grams can have at most four non-contiguous words between the two bi-gram sentences.

Example: Here is a simple ROUGE-SU* example: Suppose we have generated summary and reference summary of a climate change report.

Reference Summary:

"Global warming is a pressing issue affecting our planet.

Generated Summary:

"The issue of global warming is a concern for our plane."

Let us assume that the ROUGE-SU* implementation is aware of skip-bigrams with a maximum skip value of 2.

Suppose we find the following:

The number of overlapping skip-bigrams and unigrams: 12

The total number of skip-bigrams and unigrams in the generated summary: 25

The total number of unigrams and skip-bigrams of the reference summary: 20

i. Precision(P):

Precision (P) = Number of overlapping unigrams & skip bigrams / Total number of unigrams & skip bigrams in the generated summary

$$\text{Precision (P)} = 12 / 25 = 0.48$$

ii. Recall(R):

Recall (R) = No. of overlapping unigrams & skip bigrams / Total no. of unigrams & skip bigrams in the reference summary

$$\text{Recall (R)} = 12 / 25 = 0.6$$

iii. F1-score:

$$F1 - \text{Measure} = (2 * \text{Precision} * \text{Recall}) / (\text{Precision} + \text{Recall})$$

$$F1 - \text{Measure} = (2 * 0.48 * 0.6) / (0.48 + 0.6) = 0.536$$

VIII. RESULT AND ANALYSIS

In this section, the experimental results of the proposed extractive multi-document summarization system are discussed. The main aim of this section is to thoroughly analyze the performance of the suggested model and assess its effectiveness through widely accepted evaluation standards. The evaluation process uses ROUGE metrics, which measures how closely the summaries produced by the system match the reference summaries.

8.1 Dataset: To support continued research in automatic summarization, the National Institute of Standards and Technology (NIST) created the Document Understanding Conference (DUC), which developed the DUC-2003 dataset used in this study. The DUC-2003 dataset includes several groups of documents, each with ten related news stories that share a common topic. Each group also has human-made summaries that are considered the best examples for evaluating how well summaries perform. These summaries are all around 100 words long, which makes it easier to compare different summarization methods fairly. Because of its clear structure and the availability of these reference summaries, DUC-2003 is commonly used as a standard dataset in research related to multi-document summarization.

8.2 Evaluation of the proposed algorithm: Known automatic assessment methods are used to check how well the proposed summarizing system works. Since accuracy alone doesn't show how good a summary is, the study uses similarity-based measures instead. In this study, the system's generated summaries are checked for effectiveness using ROUGE metrics, which focus on recall. This process compares the system's summaries with ones written by people.

Table 1 Results of Proposed TLBO Algorithm

Epoch		Method	Recall	Precision	F1-Score
10	0	rouge-1	0.31944444	0.28750000	0.30263157
10	1	rouge-2	0.16129032	0.15463918	0.15789473
10	2	rouge-1	0.31944444	0.28750000	0.30263157
20	0	rouge-1	0.39189189	0.31868132	0.35151515
20	1	rouge-2	0.11956522	0.11000000	0.11458333
20	2	rouge-1	0.37837838	0.30769231	0.33939393
30	0	rouge-1	0.35714286	0.31250000	0.33333333
30	1	rouge-2	0.16129032	0.15463918	0.15789473
30	2	rouge-1	0.32857143	0.28750000	0.30666666
80	0	rouge-1	0.34146341	0.36363636	0.35220125
80	1	rouge-2	0.16129032	0.15463918	0.15789473
80	2	rouge-1	0.31944444	0.28750000	0.30263157
90	0	rouge-1	0.31944444	0.28750000	0.30263157
90	1	rouge-2	0.16129032	0.15463918	0.15789473
90	2	rouge-1	0.31944444	0.28750000	0.30263157
100	0	rouge-1	0.31944444	0.28750000	0.30263157
100	1	rouge-2	0.16129032	0.15463918	0.15789473
100	2	rouge-1	0.31944444	0.28750000	0.30263157

8.3 Comparison between the proposed algorithm and other: The DUC 2003 dataset was utilized to assess and validate the efficacy of the proposed extractive multi-document summarization algorithm. This dataset comprises 90 document clusters, with each cluster containing multiple news focused on the same subject, accompanied by human-authored reference summaries. The algorithm was executed on each cluster to produce a 100-word system summary.

To illustrate its effectiveness, the proposed method was benchmarked against several renowned optimization-based and statistical summarization techniques, such as the Firefly Algorithm, Binary Particle Swarm Optimization, Binary Genetic Algorithm and Ant Colony Optimization. These algorithms are frequently employed for sentence selection and optimization in extractive summarization tasks.

The evaluation of all methods used standard ROUGE metrics – ROUGE-1, ROUGE-2 and ROUGE-L – to check how similar the generated summaries were to the reference ones. These metrics look at single-word matches, pairs of words, and the longest shared sequence of words. The results show that the new method got better scores on all three metrics compared to other methods, showing that it is effective at choosing sentences that are relevant, informative, and not repetitive. As a result, the summaries produced are more coherent, cover more content, and are generally of higher quality.

Table 2 Parameters Used for Proposed TLBO Algorithm

Parameters for Proposed TLBO Algorithm	Value
Population Size	50
Number of Iterations	100
Teaching Factor (TF)	1 or 2
Random Number (r)	[0,1]
Summary Length Constraint	100 words

Table 3 Performance comparison of Proposed TLBO algorithm with ROUGE-1 on DUC-2003

Algorithm	ROUGE-1 (F1)
Firefly	0.325
BBO	0.315
ACO	0.31
Binary PSO	0.33
Binary GA	0.315
TLBO (Proposed)	0.3522

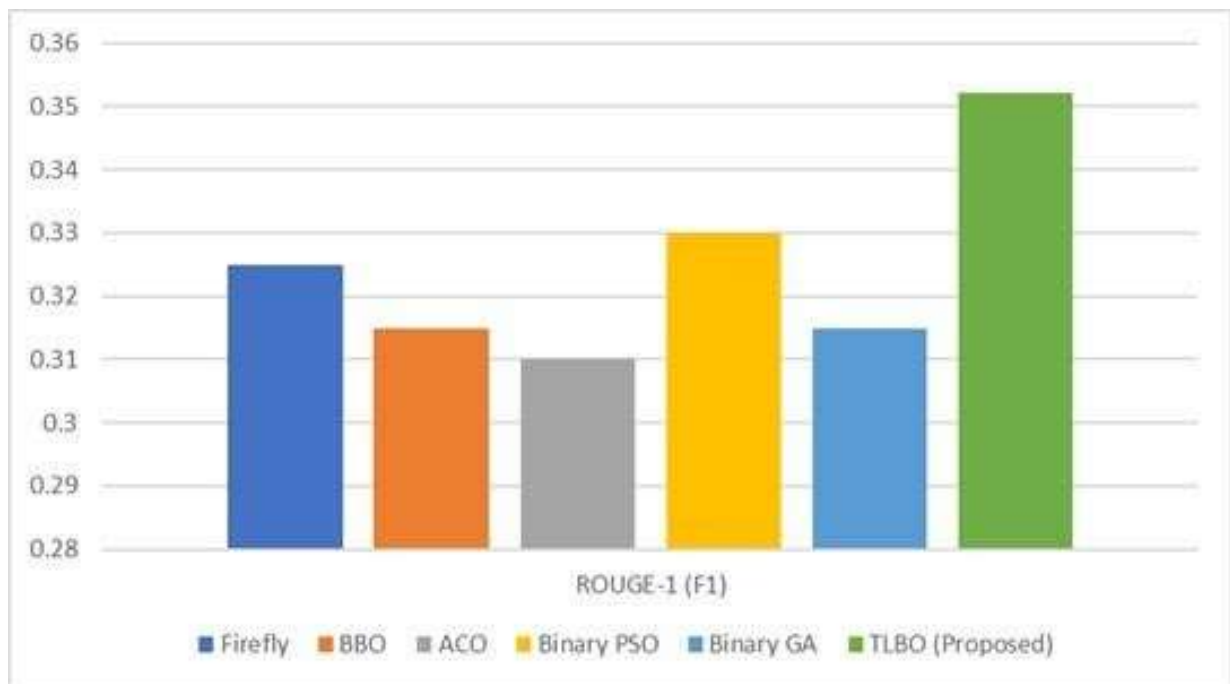


Fig.1 ROUGE-1 (F1) score comparison of proposed TLBO algorithm with other methods on DUC-2003 dataset.

Table 4 Performance comparison of Proposed TLBO algorithm with ROUGE-2 on DUC-2003

Algorithm	ROUGE-2 (F1)
Firefly	0.135
BBO	0.125
ACO	0.12
Binary PSO	0.15
Binary GA	0.13
TLBO (Proposed)	0.1579

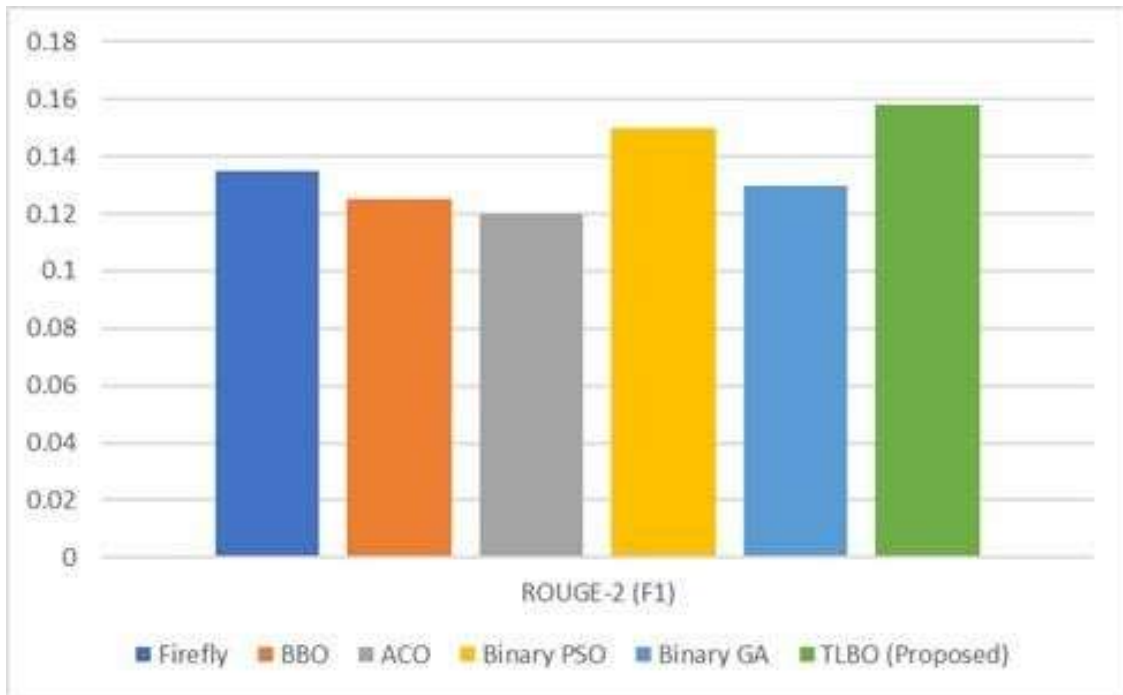


Fig.2 ROUGE-2 (F1) score comparison of proposed TLBO algorithm with other methods on DUC-2003 dataset.

Table 5 Performance comparison of Proposed TLBO algorithm with ROUGE-L on DUC-2003

Algorithm	ROUGE-L (F1)
Firefly	0.29
BBO	0.28
ACO	0.275
Binary PSO	0.3
Binary GA	0.285
TLBO (Proposed)	0.3067

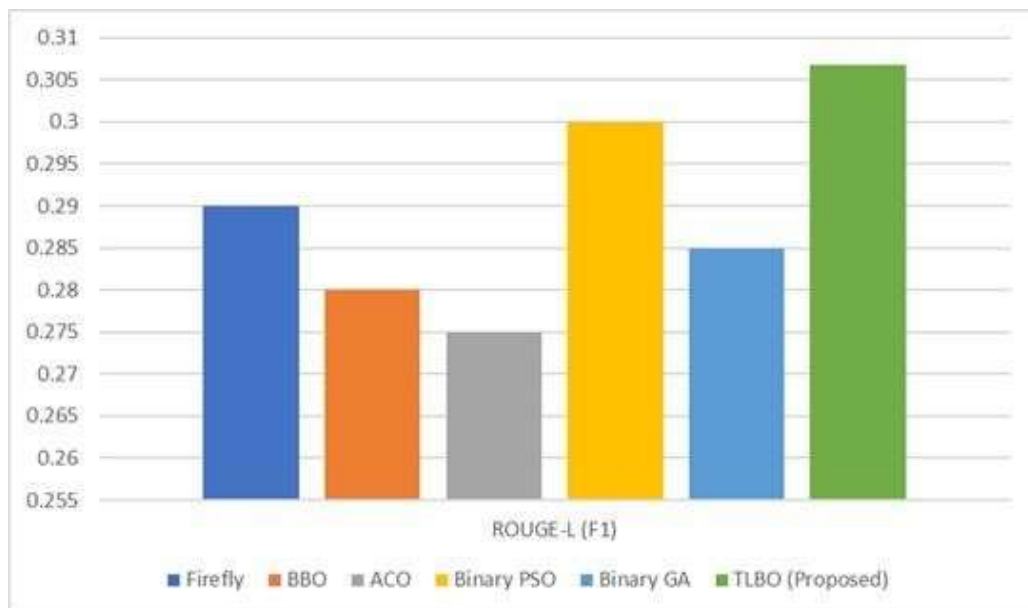


Fig.3 ROUGE-L (F1) score comparison of proposed TLBO algorithm with other methods on DUC-2003 dataset.

Table 6 Performance comparison of Proposed TLBO algorithm with ROUGE-L on DUC-2003

Algorithm	ROUGE-1	ROUGE-2	ROUGE-L
Firefly	0.325	0.135	0.29
BBO	0.315	0.125	0.28
ACO	0.31	0.12	0.275
Binary PSO	0.33	0.15	0.3
Binary GA	0.315	0.13	0.285
TLBO (Proposed)	0.3522	0.1579	0.3067

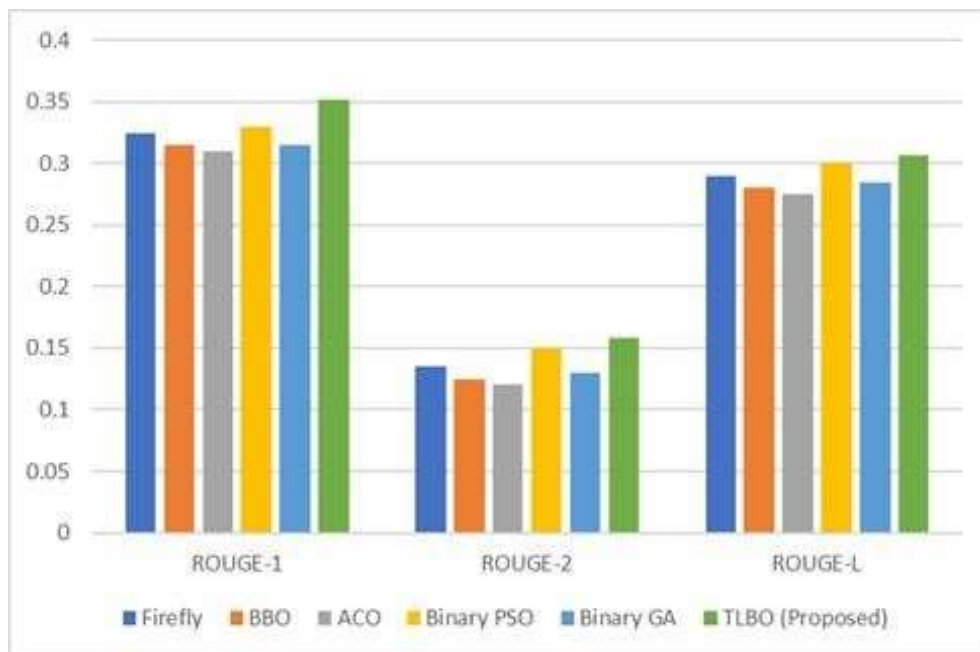


Fig.4 ROUGE (F1) score comparison of proposed TLBO algorithm with other methods on DUC-2003 dataset.

IX. FUTURE SCOPE

To improve how summaries understand context while keeping unnecessary repetition in check, future studies on extractive multi-document summarization should focus on mixed approaches that bring together transformer models like BERT and metaheuristic optimization methods. It's still important to work on making these methods scalable and less computationally heavy, especially for use in real-time and large-scale situations. Another important direction is to apply these metaheuristic methods to languages with fewer resources and to multiple languages. Also, including summarization goals that are based on user queries and personalized needs could make the system more useful and tailored to individuals. Finally, using more advanced evaluation methods beyond ROUGH, such as those that look at meaning and human feedback, would allow for a better assessment of summary quality.

X. CONCLUSION

Extractive text summarization based on metaheuristics turns out to be a potent way to deal with the constantly increasing amount of digital data. By leveraging algorithms such as Firefly, Frog Leaping, and Teaching-Learning-Based optimization, these approaches effectively balance relevance, coherence, and coverage while minimizing redundancy. Compared to conventional methods, they demonstrate superior performance on benchmark datasets and hold strong potential for multilingual, domain-specific, and real-time applications. Hence, metaheuristic-driven summarization stands as a promising direction for advancing efficient and intelligent information retrieval in NLP.

REFERENCES

- [1] Vilca, G. C. V., & Cabezudo, M. a. S. (2017b). A study of abstractive summarization using semantic representations and discourse level information. In *Lecture notes in computer science* (pp. 482–490). https://doi.org/10.1007/978-3-319-64206-2_54
- [2] El-Kassas, W. S., Salama, C. R., Rafea, A. A., & Mohamed, H. K. (2020). Automatic text summarization: A comprehensive survey. *Expert Systems With Applications*, 165, 113679. <https://doi.org/10.1016/j.eswa.2020.113679>
- [3] Sri, S. H. B., & Dutta, S. R. (2021b). A survey on Automatic text Summarization Techniques. *Journal of Physics Conference Series*, 2040(1), 012044. <https://doi.org/10.1088/1742-6596/2040/1/012044>
- [4] Patel, V., & Tabrizi, N. (2022). An automatic text summarization: a systematic review. *Computación Y Sistemas*, 26(3). <https://doi.org/10.13053/cys-26-3-4347>
- [5] Sharma, G., & Sharma, D. (2022). Automatic Text Summarization Methods: A Comprehensive Review. *SN Computer Science*, 4(1). <https://doi.org/10.1007/s42979-022-01446-w>
- [6] Bhat, I. K., Mohd, M., & Hashmy, R. (2017b). SUMITUP: a Hybrid Single-Document text Summarizer. In *Advances in intelligent systems and computing* (pp. 619–634). https://doi.org/10.1007/978-981-10-5687-1_56
- [7] I. K. Bhat, M. Mohd, and R. Hashmy, “SumItUp: A hybrid single-document text summarizer,” *Adv. Intell. Syst. Comput.*, vol. 583, pp. 619–634, 2018, doi: 10.1007/978-981-10-5687-1_56.
- [8] F. Démoncourt, M. Ghassemi, and W. Chang, “A repository of corpora for summarization,” in *Proc. 11th Int. Conf. Lang. Resour. Eval. (LREC)*, 2018, pp. 3221–3227.
- [9] K. Woodsend and M. Lapata, “Automatic generation of story highlights,” in *Proc. Annu. Meet. Assoc. Comput. Linguist.*, Jul. 2010, pp. 565–574.
- [10] A. Tandel, B. Modi, P. Gupta, S. Wagle, and S. Khedkar, “Multi-document text summarization – a survey,” in *Proc. Int. Conf. Data Min. Adv. Comput. (SAPIENCE)*, 2016, pp. 331–334, doi: 10.1109/SAPIENCE.2016.7684115.
- [11] V. Gupta and G. S. Lehal, “A survey of text summarization extractive techniques,” *J. Emerg. Technol. Web Intell.*, vol. 2, no. 3, pp. 258–268, 2010, doi: 10.4304/jetwi.2.3.258-268.
- [12] A. A. Bichi, R. Samsudin, R. Hassan, L. R. A. Hasan, and A. A. Rogo, “Graph-based extractive text summarization method for Hausa text,” *PLOS one*, vol. 18, no. 5, pp. 1–15, 2023, doi: 10.1371/journal.pone.0285376.
- [13] B. Altinel and M. C. Ganiz, “Semantic text classification: A survey of past and recent advances,” *Inf. Process. Manag.*, vol. 54, no. 6, pp. 1129–1153, 2018, doi: 10.1016/j.ipm.2018.08.001.
- [14] J. M. Sanchez-Gomez, M. A. Vega-Rodríguez, and C. J. Pérez, “A decomposition-based multi-objective optimization approach for extractive multi-document text summarization,” *Appl. Soft Comput. J.*, vol. 91, Jun. 2020, doi: 10.1016/j.asoc.2020.106231.
- [15] D. V. P. Kumar, S. S. Raj, P. Verma, and S. Pal, “Extractive text summarization using meta-heuristic approach,” 2022.
- [16] P. S. Premjith, A. John, and M. Wilscy, “Metaheuristic optimization using sentence level semantics for extractive document summarization,” 2015.
- [17] S. Bano, S. Khalid, N. M. Tairan, H. Shah, and H. A. Khattak, “Summarization of scholarly articles using BERT and BiGRU: Deep learning-based extractive approach,” *Journal of King Saud University - Computer and Information Sciences*, vol. 35, no. 9, oct. 2023, doi: 10.1016/j.jksuci.2023.101739.
- [18] I. Akhmetov, A. Gelbukh, and R. Mussabayev, “Greedy optimization method for extractive summarization of scientific articles,” *IEEE Access*, vol. 9, pp. 168141–168153, 2021, doi: 10.1109/ACCESS.2021.3136302.
- [19] A. M. A. Zeyad and A. Biradar, “A hybrid text summarization approach using neural networks and metaheuristic algorithms,” 2023.
- [20] A. M. A. Zeyad and A. Biradar, “A hybrid text summarization approach using neural networks and metaheuristic algorithms,” 2023.
- [21] D. Debnath, R. Das, and P. Pakray, “Extractive single document summarization using multi-objective modified cat swarm optimization approach: ESDS-MCSO,” 2001.
- [22] A. Abuobeida, “Text summarisation using swarm-based and genetic-based methods in natural language processing: A comparative review,” 2025.
- [23] D. M. Dunlavy, D. P. o’Leary, J. M. Conroy, and J. D. Schlesinger, “QCS: A system for querying, clustering and summarizing documents,” *Inf. Process. Manage.*, vol. 43, no. 6, pp. 1588–1605, 2007.
- [24] R. V. Rao, V. J. Savsani, and D. P. Vakharia, “Teaching-learning-based optimization: A novel method for constrained mechanical design optimization problems,” *CAD Comput. Aided Des.*, vol. 43, no. 3, pp. 303–315, 2011, doi: 10.1016/j.cad.2010.12.015.
- [25] M. Boussaleem, S. Aitouche, M. Hamouma, H. Haouassi, and H. Rahab, “BBMA-MDS: Binary biology migration algorithm for multi-document text summarization,” *Revue d’Intelligence Artificielle*, Nov. 2023, doi: 10.18280/ria.370506.
- [26] M. Tomer and M. Kumar, “Multi-document extractive text summarization based on firefly algorithm,” *J. King Saud Univ. - Comput. Inf. Sci.*, vol. 34, no. 8, pp. 6057–6065, Sep. 2022, doi: 10.1016/j.jksuci.2021.04.004.
- [27] A. Zhou, B. Y. Qu, H. Li, S. Z. Zhao, P. N. Suganthan, and Q. Zhang, “Multiobjective evolutionary algorithms: A survey of the state of the art,” *Swarm Evol. Comput.*, vol. 1, no. 1, pp. 32–49, 2011, doi: 10.1016/j.swevo.2011.03.001.
- [28] E. Lloret, L. Plaza, and A. Aker, “The challenging task of summary evaluation: An overview,” *Lang. Resour. Eval.*, vol. 52, no. 1, pp. 101–148, 2018, doi: 10.1007/s10579-017-9399-2.

- [29] N. K. Nagwani and S. Verma, "A Frequent Term and Semantic Similarity based Single Document Text Summarization Algorithm," *International Journal of Computer Applications*, vol. 17, no. 2, pp. 36–40, Mar. 2011, doi: 10.5120/2190-2778.
- [30] N. K. Nagwani and S. Verma, "A Frequent Term and Semantic Similarity based Single Document Text Summarization Algorithm," *International Journal of Computer Applications*, vol. 17, no. 2, pp. 36–40, Mar. 2011, doi: 10.5120/2190-2778.
- [31] N. K. Nagwani and S. Verma, "A Frequent Term and Semantic Similarity based Single Document Text Summarization Algorithm," *International Journal of Computer Applications*, vol. 17, no. 2, pp. 36–40, Mar. 2011, doi: 10.5120/2190-2778.
- [32] M. Tomer and M. Kumar, "Multi-document extractive text summarization based on firefly algorithm," *Journal of King Saud University – Computer and Information Sciences*, vol. 34, no. 8, pp. 6057–6065, 2022, doi: 10.1016/j.jksuci.2021.04.004.
- [33] M. Boussaleem, S. Aitouche, M. Hamouma, H. Haouassi, and A. Bekhouche, "BBMA-MDS: Binary Biology Migration Algorithm for Multi-Document Text Summarization," *Revue d'Intelligence Artificielle*, vol. 37, no. 5, pp. 1147–1156, 2023.
- [34] R. Desai, K. Patel, and P. Shah, "BERT-Similarity-Based Firefly (BSBF) Algorithm for Multiple Document Summarization," *ICTACT Journal on Data Science and Machine Learning*, vol. 4, no. 4, pp. 520–525, 2023.
- [35] S. S. Aote, A. Pimpalshende, A. Potnurwar, and S. Lohi, "Binary Particle Swarm Optimization with an Improved Genetic Algorithm for Multi-Document Text Summarization," *Engineering Applications of Artificial Intelligence*, vol. 117, 2023.
- [36] M. Tomer and M. Kumar, "Enhancing metaheuristic-based extractive text summarization with fuzzy logic and advanced optimization," *Neural Computing and Applications*, 2023.
- [37] D. Tomar and M. Kumar, "An approach for extractive text summarization using evolutionary and swarm optimization algorithms," *Expert Systems with Applications*, vol. 197, 2022.
- [38] A. Agarwal, O. Mittal, and P. Bansal, "A survey of text summarization using swarm intelligence optimization algorithms," *Knowledge-Based Systems*, vol. 163, pp. 518–532, 2019.
- [39] T. Jayaraman and A. S. Senthilrajan, "Brainstorm optimization for multi-document summarization," *Turkish Journal of Computer and Mathematics Education*, vol. 12, no. 10, pp. 2992–3003, 2021.
- [40] A. Jain, R. Gupta, and S. Mehta, "Automatic text summarization using real coded genetic algorithm," *Applied Sciences*, vol. 12, no. 13, p. 6584, 2022.