FACTA UNIVERSITATIS
Series: **Electronics and Energetics** Vol. 38, N° 3, September 2025, pp. 533 - 551
https://doi.org/10.2298/FUEE2503533R

Original scientific paper

IMPROVING EXTRACTIVE TEXT SUMMARIZATION VIA EFFICIENT COATI ALGORITHM FOR SINGLE DOCUMENT

Jyotirmayee Rautaray[1], Sangram Panigrahi[2], Ajit Kumar Nayak[2]

[1]Department of Computer Science & Engineering, Siksha 'O' Anusandhan (Deemed to be University), Bhubaneswar-751030, Odisha, India
[2]Department of Computer Science & Information Technology, Siksha 'O' Anusandhan (Deemed to be University), Bhubaneswar-751030, Odisha, India

ORCID iDs: Jyotirmayee Rautaray https://orcid.org/0000-0003-2747-3919
Sangram Panigrahi https://orcid.org/0000-0003-1703-4613
Ajit Kumar Nayak https://orcid.org/0000/0003/2302-9458

Abstract. *In the digital era, the rapid expansion of online information demands efficient automated text summarization techniques to extract key insights from large documents. This study introduces a novel single-document extractive summarization approach that utilizes Term Frequency-Inverse Topic Frequency (TF-ITF) for feature extraction and the Coati Optimization Algorithm (COA) for optimal sentence selection. COA enhances summarization performance by balancing precision and recall through an adaptive fitness function, improving the quality of extracted summaries. The proposed model is evaluated on DUC 2002, 2003, and 2005 datasets using ROUGE, BLEU, precision, recall, and F1-score metrics. Comparative analysis against state-of-the-art optimization algorithms, including PSO, CSO, GWO, BCO, QABC, MCSO, and GLO, demonstrates that COA outperforms existing techniques, achieving higher recall and F1 scores while maintaining competitive precision. These findings establish COA as an effective optimization technique for enhancing automated text summarization.*

Key words: *Term Frequency-Inverse Topic Frequency, Coati Optimization Algorithm, vectors, Single Document Text Summarization, Rouge scores, BLEU score*

1. INTRODUCTION

The major aim of text summarization is to condense long written texts into a precise, concise and understandable format which highlights the most important details from the original source. By picking out important lines and incorporating all pertinent details from the source text, automatic text summarization creates summaries. In Natural Language Processing (NLP), summarization is a major challenge since it necessitates

Received December 19, 2024; revised March 01, 2025 and March 15, 2025; accepted March 17, 2025
Corresponding author: Jyotirmayee Rautaray
Department of Computer Science & Engineering, Siksha 'O' Anusandhan (Deemed to be University), Bhubaneswar-751030, Odisha, India
E-mail: jyotirmayee.1990@gmail.com

thorough text analysis, including lexical and semantic analysis, in order to generate high-quality summaries. The two primary methods of summarization are abstractive and extractive. Whereas abstractive summarization interprets and rewords the essential portions to produce the final summary, extractive summarization finds and immediately copies the most significant passages from the source material into the summary [1][2]. Creating a summary that successfully communicates the main ideas of the original text is the goal of abstractive summarization, which frequently involves merging words or phrases which are not found in the given text [3]. Conversely, extractive summarizing generates a summary solely from the original material by using the text's original words, structures, or phrases. Automated extractive text summarization is a valuable tool in education, as it efficiently extracts key elements without requiring manual effort or human intervention [4].

Depending on the type of the inputs, the documents can be divided into two categories: single-document and multi-document summarization. In contrast to multi-document summary, which entails summarizing a group of connected documents, this work concentrates on single-document text summarization, in which the input consists of a single document [5]. Summarization involves three key objectives: generating the summary from one or more documents, retaining essential information, and producing a concise summary [6]. For single-document systems, the summary is generated solely from that individual document. Four main extractive summarization techniques are commonly used, depending on the text: machine learning, meta-heuristics, statistical, and semantic methods [7][8]. In order to find near-optimal solutions for complicated problems, metaheuristic optimization algorithms use techniques modelled after natural processes, such as evolution or swarm activity, to explore the solution space. Coati Optimization, a metaheuristic approach, emulates the foraging behavior of coatis, balancing exploration and exploitation by simulating their adaptive and dynamic search strategies to find optimal solutions [9]. Recent advancements have focused on refining these models with attention mechanisms and fine-tuning on large datasets to generate more accurate and context-aware summaries [10]. Regardless of previous summaries, the main goal is to quickly create one from a given text or collection of documents using a variety of methods and algorithms. The goal of metaheuristic algorithms in this context is to identify high-scoring phrases. These methods are employed in text summaries to choose the best or nearly best collection of sentences that create an understandable and instructive synopsis. Examples include genetic algorithms and other optimization methods [11][12][13]. An innovative optimization method that draws inspiration from coatis' natural behaviors is the Coati Optimization Algorithm (COA). COA provides a number of benefits for resolving global optimization issues, including the elimination of the need for parameter adjustments due to its lack of control parameters and its high effectiveness in addressing a big range of optimization problems in different scientific domains, including intricate high-dimensional issues.

1.1. Contribution

- An organized method for summarizing a single document that includes TF-ITF feature extraction and thorough text preprocessing, improving the accuracy and applicability of the summaries.
- This paper introduces the innovative use of the COA for summary generation, optimizing vector-based processes with a unique fitness function, achieving greater efficiency compared to traditional methods.

- The study uses ROUGE scores, BLEU scores, accuracy, recall, and F-score metrics to statistically assess the efficacy of the COA on the DUC 2002, 2003, and 2005 datasets. It shows that the COA can generate clear and insightful summaries from complicated textual material.

Structure of the paper: Section 1 provides an overview of text summarization and its various forms; Section 2 reviews the literature on document summarization using various methods and algorithms; Section 3 introduces the proposed model, methods, and COA; Section 4 covers the research findings and result analysis; and Section 5 concludes the study.

2. Related Works

While multi-document summarizing entails producing a summary from several papers, single-document summarization creates a summary from a single document. While it is possible to apply single-document summary techniques to multi-document summarization, summarizing several documents is far more difficult. This section examines previous attempts in the literature on text summarization and looks at several optimization techniques and algorithms that have been put forth for this aim.

Cheng et al. [14] proposed a data-driven approach leveraging continuous sentence features and neural networks. They developed a hierarchical document-based framework to support single-document summarization. The models are trained with very big datasets large number of document-summary pairs, without relying on language annotations. Two types of models were created, focusing on word and phrase extraction. This approach enables the models to learn informativeness characteristics through continuous approximations, enhancing the summarization process.

Kryściński et al. [15] proposed a model-based, weakly-supervised approach for detecting discrepancies and verifying factual consistency between source documents and generated summaries. Sentences from source texts are modified using rule-based transformations to create training data. The model is trained on three key tasks: 1) Assessing whether sentences retain factual consistency after translation; 2) Extracting a supporting span from the source documents that upholds the consistency assumption; and 3) Identifying any incongruent spans from the summary sentence.

Debnath, D et al. [16] proposed an Archive-based Micro Genetic-2 Algorithm to tackle the multi-objective Extractive Single Document Summarization problem. The evaluation was conducted using the DUC-2001 and DUC-2002 datasets, and the results were compared with previous methods using ROUGE metrics.

Timea Bezdan et al. [17] introduced a Hybrid FFO method that outperforms K-Means for text document clustering. The case study, which examined text documents with limited functionalities, demonstrated the effectiveness of that given approach.

Debnath, D et al. [18] addressed a single-document extraction problem for automated text summarization and used Cat Swarm Optimization (CSO). CSO aims to produce useful, redundant-free summaries with ample coverage. Compared to the leading dataset techniques, ROUGE-1 and ROUGE-2 scores improved by 25% and 5%, respectively.

Pati and Rautray et al. [19] employed the DUC 2003 dataset to showcase the superior performance of Cuckoo Search (CS) for single-document extractive summarization, comparing it with the Firefly Algorithm (FFA) and Ant Colony Optimization (ACO).

Svore et al. [20] introduced NetSum, a novel automated summarization method utilizing neural networks. In this approach, each sentence is analyzed based on a set of features that highlight its importance within the text. The method incorporates advanced features derived from Wikipedia entities and recent search query data.

Mandal, S et al. [21] proposed a method combining sentiment analysis, language scoring, and Cuckoo Search (CS) computation. The approach uses sentence scoring techniques to evaluate phrases based on mathematical frameworks, and CS computation is then applied to select the most suitable phrases for generating the summary.

Jain et al. [22] proposed using the PSO algorithm for text summarization in the Punjabi language. The search process is conducted by rapidly moving particles that update their positions and velocities at the end of each iteration. Throughout the generations, the algorithm continuously updates the personal best and global best solutions.

Zhang et al. [33] proposed a comprehensive survey on text summarization, transitioning from statistical methods to large language models (LLMs). It reviewed advancements in benchmarking, modeling, and evaluation metrics, emphasizing the role of pre-trained language models (PLMs) and LLMs in improving summarization tasks. The study utilized various standard datasets to evaluate the techniques and provided insights into the latest trends and challenges.

Yadav et al. [34] proposed an analysis of extractive and abstractive text summarization techniques to address information overload. The study explored standard datasets, evaluation metrics, and highlighted challenges in creating advanced summarizers. It reviewed techniques such as extractive and abstractive summarization and analyzed their effectiveness on widely used summarization datasets.

Mirjalili et al. [35] proposed the Grey Wolf Optimizer (GWO), an optimization algorithm inspired by the hierarchical hunting mechanism of grey wolves. GWO has been applied in text summarization to enhance sentence selection based on relevance and informativeness. The algorithm effectively balances exploration and exploitation, leading to high-quality summaries. However, it may suffer from premature convergence in high-dimensional datasets.

Karaboga et al. [36] introduced the Bee Colony Optimization (BCO) algorithm, which mimics the foraging behavior of honeybees to extract key sentences for summarization. This method effectively reduces redundancy and enhances informativeness by leveraging swarm intelligence. Nevertheless, its performance is highly dependent on parameter tuning, which can impact consistency across different datasets.

Wang et al. [37] developed Multi-Colony Swarm Optimization (MCSO) for text summarization, where multiple cooperating colonies work together to extract meaningful sentences. This approach enhances the diversity and quality of generated summaries through multi-objective optimization. However, the increased computational complexity due to interactions among multiple colonies can be a drawback for large-scale document processing.

Sharma et al. [38] proposed Glowworm Swarm Optimization (GLO) for extractive summarization. Inspired by glowworm luminescence, this algorithm dynamically selects relevant sentences based on a luciferin-based attraction mechanism. The adaptability of GLO ensures high-quality summaries with strong contextual relevance. However, it can face computational overhead when processing large document sets.

Yuan et al. [39] introduced the Quick Artificial Bee Colony (QABC) algorithm, an enhanced version of the traditional BCO designed for faster convergence in optimization problems, including text summarization. By refining the search mechanism and reducing

unnecessary computations, QABC improves search efficiency and sentence selection speed. However, it still requires careful parameter tuning to maintain robustness across diverse datasets.

Table 1 Existing research contains a range of optimization techniques

Sl. no	Author, Reference	Dataset	Methodology	Advantage	Disadvantage
1	Cheng et al., [14]	DUC 2002, Daily Mail news highlights corpus	Encoding and attention-based extractor	The approach leverages the power of neural networks for more effective summarization without requiring hand-crafted features	Data Dependency, Relies on Extractive Summarization, Complexity
2	Debnath D et al., [16]	DUC 2002, DUC 2001	AMGA2	Efficient for extractive summarization with multi-objective optimization.	Computationally expensive - Performance depends on parameter tuning.
3	Kryściński et al., [15]	CNN/ DailyMail	BERT	Leverages pre-trained transformers for high-quality abstractive summaries.	Struggles with factual consistency in longer documents.
4	TimeaBezdan et al., [17]	Text datasets	FFA	Good for feature selection, improving summary relevance and quality.	Not suitable for highly dynamic or complex datasets.
5	Debnath D et al., [18]	DUC 2002, DUC 2001	CSO	Enhances coherence and accuracy in extractive summarization tasks.	Limited generalization across diverse summarization datasets.
6	Pati and Rautray, et al., [19]	DUC 2003	ACO, FFA, and CSO	Hybrid approach improves efficiency, accuracy, and feature optimization.	Increased model complexity and resource requirements.
7	Svore et al., [20]	DUC 2002, DUC 2003	Rank Net learning algorithm	Scalability, Relevance Ranking	Generalization Issues, Potential for Information Overload
8	Mandal S et al., [21]	Kaggle dataset	CSA	Incorporation of Sentiment Analysis, Scalability, Feature Integration	Generalizability Issues, Dataset Dependency, Computational Complexity
9	Jain et al., [22]	Punjabi datasets	PSO	Optimization Efficiency, Feature-Based Scoring, Scalability	Language Dependency, Limited Dataset, Lack of Semantic Understanding

10	Zhang et al., [33]	Various summarization datasets	Statistical, Deep Learning, and LLMs	Offers a thorough historical and contemporary analysis of text summarization methods	Does not propose new models or techniques and relies heavily on existing literature.
11	Yadav et al., [34]	Standard summarization datasets	Text Rank, Seq2Seq	Offers a comprehensive overview of state-of-the-art methods, aiding researchers in understanding advancements in the field.	Heavily relies on existing datasets and benchmarks, limiting novelty
12	Mirjalili et al., [35]	Standard benchmark datasets for optimization problems	Grey Wolf Optimizer (GWO)	Balances exploration and exploitation efficiently, leading to high-quality summaries	May converge prematurely in complex, high-dimensional problems
13	Karaboga et al., [36]	Various text datasets, including news articles	BCO	Reduces redundancy and enhances informativeness through swarm intelligence	Performance highly dependent on parameter tuning
14	Wang et al., [37]	DUC datasets (DUC-2001, DUC-2002)	MCSO	Multi-objective optimization ensures diverse and high-quality summaries	Increased computational complexity due to multiple colony interactions
15	Sharma et al., [38]	Scientific and news article datasets	GLO	Dynamically adjusts selection based on informativeness and context	May struggle with large document sets due to computational overhead
16	Yuan et al., [39]	Summarization benchmark datasets	QABC	Improves search efficiency and sentence selection speed through refined search mechanisms	Requires careful parameter tuning to maintain robustness across datasets

Existing text summarization methods encounter several challenges, including difficulty in generalizing to various document structures and maintaining factual consistency between summaries and source texts. Techniques such as Genetic Algorithms and Firefly Algorithms often fall short in multi-objective optimization, while neural networks and sentiment analysis approaches may struggle to adapt to diverse text types and languages. Additionally, Particle Swarm Optimization-based methods may prove inadequate for handling complex summarization tasks effectively. The COA addresses these limitations by balancing exploration and exploitation, which enhances adaptability to various text types and structures. COA improves search efficiency for optimal summarization solutions and reduces reliance on extensive datasets, making it more effective in multi-objective problems and improving overall summarization accuracy.

The surge of vast electronic texts in the digital age has created a growing need for efficient automated text summarization methods to distill essential information succinctly. Current

extractive and abstractive approaches often struggle to accurately capture key content while maintaining readability and coherence. Moreover, many existing models depend on complex linguistic annotations and manual feature engineering, which can hinder scalability and adaptability across various text types. By investigating neural network-based methods and optimization algorithms, this study tries to address above issues and uplift the effectiveness and caliber of single-document summarization. The goal is to create strong frameworks that generate excellent summaries without the need for a lot of human input or language resources.

3. Proposed Method

The approach follows a structured workflow designed to generate concise and informative summaries effectively. The process begins with text preprocessing, including cleaning, tokenization, stop word removal, and lemmatization to standardize the content. In the feature extraction phase, words are transformed into vector representations, and sentence relevance is ranked using TF-IDF. Additional processing is applied to refine vector features. During summary generation, the COA selects the most informative sentences using a fitness function. The final summary is then evaluated using metrics such as ROUGE score, BLEU score, precision, recall, and F-score. These steps are visually represented in the improved Figure 1.

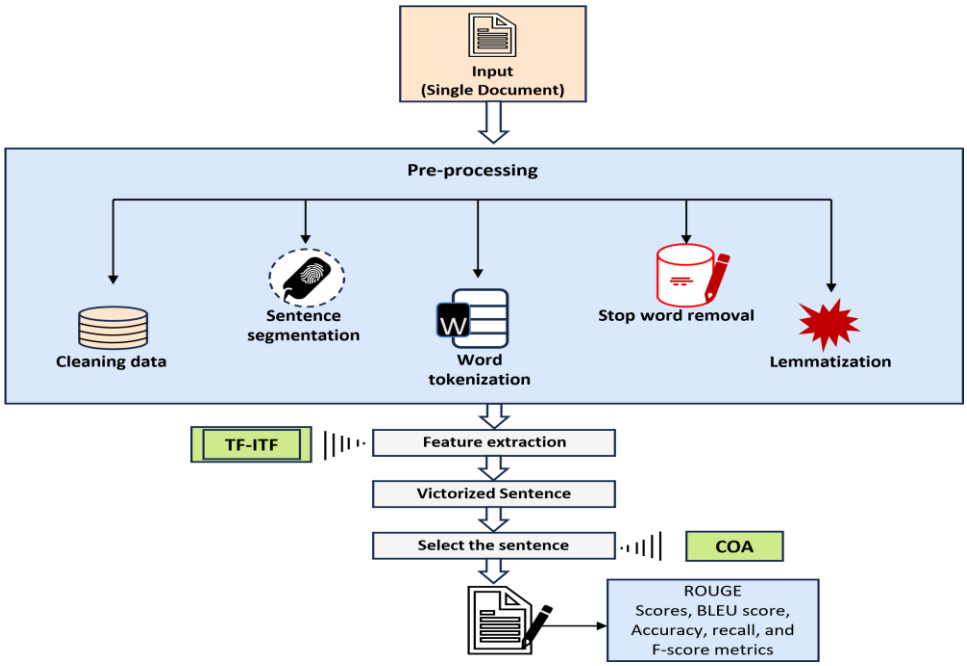

Fig. 1 Flow Diagram of Single Document Summarization

3.1. Text pre-processing

This procedure, which comes before summary, entails transforming the original report into a more organized and controllable data format. To summarize individual documents, the DUC

2002, DUC 2003, and DUC 2005 databases are utilized. These datasets are widely recommended benchmarks in text summarization research. These datasets offer high-quality, manually curated summaries that ensure a rigorous evaluation of summarization techniques. While modern datasets like Reddit, the New York Times Annotated Corpus, or WikiNow are relevant for contemporary challenges, the DUC datasets remain a preferred choice due to their structured nature and established use in benchmarking.

Segmenting sentences, tokenizing words, eliminating stop words, and lemmatizing words are important steps in this process.

Cleaning data: To clean data, first identify and handle missing values by removing or imputing them. Next, duplicate entries are removed to ensure data consistency. Finally, text data is standardized by converting to lowercase and stripping whitespace.
Sentence Segmentation: Sentence segmentation entails tokenizing the individual words that make up sentences. Punctuation, including commas, semicolons, question marks, colons, and periods is used to divide the message into sentences [23].
Word Tokenization: Tokenization divides sentence onto words according to grammar and blank spaces [23].
Stop Word Removal: These are those words that carry little to no significant meaning, such as conjunctions, articles, possessive words, pronouns, and relational terms. These words, like "is," "and," and "the," can negatively impact the efficiency of processing large tokens, making it essential to remove them from text during analysis. After dividing the text into paragraphs, these stop words are filtered out to improve the relevance of the remaining words [24].
Lemmatization: Lemmatization is the process of reducing words to their root words in order to lessen their redundancy [25].
The basic steps involved in text preprocessing steps are illustrated in figure 2 given below.

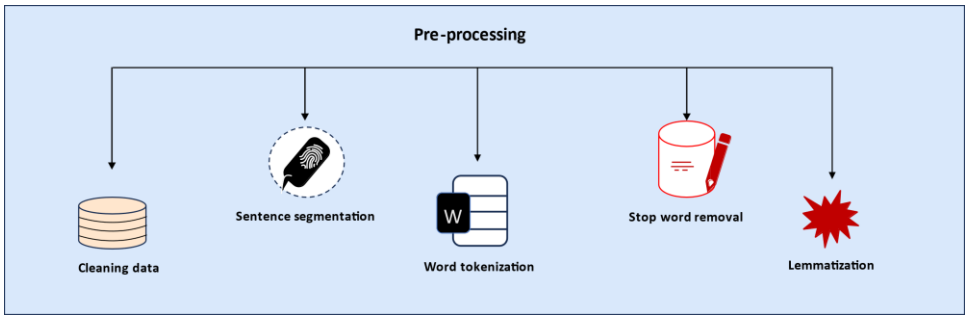

Fig. 2 Overview of Text Preprocessing Steps

3.2. Feature Extraction

A numerical statistical technique called Term Frequency-Inverse Topic Frequency (TF-ITF) is applied in NLP applications including data extraction and text mining. It enhances the traditional Bag of Words method for converting text into vectors by considering a word's significance within a specific document relative to other documents in the corpus. A word's TF-ITF score is calculated by multiplying two statistical components. The first, term frequency (TF), measures how important a word is within a particular document. The second,

inverse topic frequency (ITF), reflects how frequently the word appears across the entire corpus. As a result, words that occur frequently in all documents receive lower scores. The TF-ITF output for each document is a high-dimensional sparse vector, where the number of non-zero elements correspond to the count of unique words in the text as shown in equ[n] (1), (2), and (3) [26].

$$TF\left(term\right)=\frac{frequency\ of\ \ term\ in\ the\ document}{Total\ \ no.\ of\ \ terms\ in\ the\ document} \tag{1}$$

$$ITF\left(term\right)=\log\frac{Total\ \ no.\ of\ \ topic}{no.\ of\ \ topic\ with\ that\ terms\ in\ it} \tag{2}$$

$$TF-ITF\left(term\right)=TF\left(term\right)*ITF\left(term\right) \tag{3}$$

TF-ITF values range between [0,1] with ten-digit precision. Once these values are calculated, the terms are arranged in descending order. Each term is then paired with its respective value to create a new word scenario. This arrangement is crucial for analyzing the TF-ITF values of individual words, allowing for the examination of previously overlooked results. The significance of a phrase is determined by calculating the TF-ITF value of each word, and the overall importance of the phrase is derived from the combined value of all words, including the action word. The words are then listed chronologically in descending order of their importance. The TF-ITF technique follows the traditional TF-IDF principles, where word importance is determined based on document frequency. Essentially, TF-ITF is conceptually the same as TF-IDF, and we acknowledge that the standard term "Inverse Document Frequency (IDF)" should be used for consistency.

Tf_ITF calculation with one example is shown through the given example. Consider a corpus with **5 topics**, and a document containing the following words with respective frequencies:

Term	Frequency in Document	Total Terms in Document	Topics Containing Term
“cancer”	4	100	3
“scan”	2	100	2
“deep”	1	100	5

TF("cancer") =100/4=0.04
TF("scan") =100/2=0.02
TF("deep") =100/1=0.01
ITF("cancer") = log (5/3)=0.22
ITF("scan") = log (5/2)=0.40
ITF("deep") = log (5/5)=0.00
TF_ITF("cancer") = 0.04×0.22=0.0088
TF_ITF("scan") = 0.02×0.40=0.0080
TF_ITF("deep") = 0.01×0.00=0.0000

After computing these values, the terms are ranked in descending order of importance:
1. "cancer" → 0.0088
2. "scan" → 0.0080
3. "deep" → 0.0000

3.3. Vectorization

The phrases are now transformed into vectors in this stage. Each phrase is broken up into a list of separate words. Since every word in the collection has a TF-ITF score, it is allocated to them. The words' probable vector forms are listed in this list of TF-ITF scores. The algorithm then receives these vectors in order to process and produce an outcome [27]. In our proposed approach, words are represented using TF-ITF, a numerical statistical technique for feature extraction in text processing. Unlike word embeddings, which capture semantic relationships between words, TF-ITF focuses on statistical significance by determining a word's importance within a document relative to a corpus. This means our method does not rely on contextual similarity or distributional semantics but instead emphasizes the frequency-based importance of words. Sentences are then represented as vectors of TF-ITF weights, allowing for effective text representation without requiring word embeddings. These TF-ITF-based vectors are then used as input for further processing in our vectorization stage, where each phrase is broken into separate words and assigned their respective TF-ITF scores. The resulting vectors are then optimized using the COA to enhance the performance of the model, ensuring effective parameter tuning for improved classification accuracy.

By using the COA to choose hyperparameters such Learning Rate, Batch Size, Dropout Rate, and Embedding Dimension optimally, they improve the vectorization. This approach ensures efficient and effective parameter tuning.

3.3.1. Hyper-parameter optimization using Coati Optimization Algorithm (COA)

Coatis, also called coatimundis, belong to the Procyonidae family's Nasua and Nasuella genera, which belong to diurnal animal. Each coati has a long, non-prehensile tail used for balance and signalling, black paws, tiny ears, and a slender head with a flexible, elongated, somewhat upward-turned nose. The adult coatis can be as long as their body, measuring between 33 to 69 cm from top to bottom tip [28]. COATI optimization algorithm is used for improving extractive summarization rather than deep learning-based models. Unlike neural summarization models, which require large-scale training data and significant computational resources, COATI provides an efficient and interpretable optimization technique that enhances summarization outcomes without extensive learning-based mechanisms.

The COATI optimization algorithm is inspired by coatis' natural hunting and escape behaviors. During the **hunting phase**, coatis search for food by exploring various locations, which mirrors the algorithm's **global search process**—broadly exploring the solution space to identify optimal parameters. In the **escape phase**, coatis swiftly adjust their positions to evade predators, resembling the **local search phase**, where the algorithm fine-tunes its parameters for better optimization. By integrating these two strategies, COATI efficiently optimizes hyperparameters, enhancing extractive summarization performance. Hyper-parameter optimization is the process of determining the best mix of vectorization hyper-parameter settings to optimize performance on data in a reasonable quantity. This process is essential to vectorization capacity for precise result prediction. Most of this input text uses the hyperparameters' default values. The proposed model optimizes the hyperparameter utilizing the COA. The hyperparameter values for learning rate, batch size, and dropout rate are selected based on common practices to balance model performance and efficiency. The learning rate, typically ranging from 10^{-5} *to* 10^{-1}, is chosen to ensure stable convergence; too small a value slows learning, while too large a value can lead to instability. Batch sizes of 16, 32, 64, 128, and 256 are used to balance computational efficiency and generalization, with

smaller sizes offering noisy but beneficial gradient updates, and larger sizes providing stable gradients but requiring more memory. The dropout rate, ranging from 0.1 to 0.5, helps prevent overfitting by randomly deactivating neurons during training, where lower values provide minimal regularization and higher values offer stronger regularization to ensure robust learning. These ranges are widely used because they offer flexibility in achieving an optimal model configuration.

The COA is used to optimize these hyperparameters. The COA step-by-step procedure is explained below.

Step 1: Initialization: The main idea behind this method is to catch the optimal hyperparameter. First, establish the problem's upper and lower boundaries, the variables' dimensionality D, the maximum number of iterations, and the Coati size N. LR, BS, DR, and ED are among the hyperparameters that make up each solution that the Coati represents. First, a selection is made at random. The following equation displays the initial solution format:

$$P_N = \{S_1, S_2, \ldots S_N\} \tag{4}$$

Here, P_N is the Nth solution or Coati's position

$$S_i = \{LR, BS, DR, ED\}_i \tag{5}$$

Step 2: Fitness calculation: After initialization, each solution's fitness is evaluated using the suggested AO2 technique. In this instance, the fitness function is used to define the classification accuracy. The most effective solution is one with the topmost fitness value. The fitness function is determined with the help of the equation:

$$Fitness = Max\left(\frac{TP+TN}{TP+TN+FP+FN}\right) \tag{6}$$

Step 3: Updating using COA: COA utilizes 2 distinct techniques known as the attacking and hunting strategy on iguanas and process of escaping from predators.

Strategy 1: Hunting and attacking strategy on iguana

Coatis moves around in the search space as a result of this strategy, demonstrating the COA's ability to do global research within the problem-solving domain.

$$S_i^{P1} : S_{i,j}^{P1} = S_{i,j} + r.\left((Iguana_j - I.S_{i,j})\right) \tag{7}$$

After reaching the floor, the iguana is placed at random around the search area. Coats on the ground move in the search space based on this random placement, where N is the number of coatis.

$$Iguana^G : Iguana_j^G = I_{Z_j} + r.(uz_j - lz_j), j = 1, 2, \ldots, m \tag{8}$$

$$S_i^{P1} : S_{i,j}^{P1} = \begin{cases} S_{i,j} + r.(Iguana_j^G - I, S_{i,j}), & F_{Iguana^G < F_1} \\ S_{i,j} + r.(S_{i,j} + Iguana_j^G) & else \end{cases} \tag{9}$$

$$For\ i = \left\lfloor \frac{N}{2} \right\rfloor + 1, \left\lfloor \frac{N}{2} \right\rfloor + 2, \ldots, N \tag{10}$$

The update mechanism accepts the new position that is determined for each coati when it raises the value of the target function; otherwise, the coati remains in its original location. This update need is intended for i = 1, 2, . . . , N.

$$S_i = \begin{cases} S_i^{P1}, F_i^{P1} < F_i \\ S_i, \ else \end{cases}, \tag{11}$$

Here S_i^{P1} represents the newly calculated location for the i[th] coati, while $S_{i,j}^{P1}$ denotes its j[th] measurement, F_i^{P1} is the value of its objective function. The Iguana indicates the location of the best performing member in search region, with Iguana representing its jth dimension.$Iguana_j^G$ refers to the jth measurement at this randomly chosen position, while F_{Iguana}^G represents the value of the objective function at this location.

Strategy 2: The act of running away from a predator

Because of the maneuvers it has made in this technique, Coatis'position is secure with relation to its present position, which implies that the COA can be employed in local search.

$$lz_j^{local} = \frac{lz_j}{t}, uz_j^{local} = \frac{lz_j}{t}, where\ t = 1, 2, \ldots, T \tag{12}$$

$$S_i^{P2} : S_{i,j}^{P2} = S_{i,j} + (1-2r).\left(lz_j^{local} + r.(uz_j^{local} + r.(uz_j^{local} - lz_j^{local}))\right) \tag{13}$$

The newly found and determined position is found to be appropriate if it raises the objective function's value.

$$S_i = \begin{cases} S_i^{P2}, F_i^{P2} < F_i \\ S_i, \ else \end{cases}, \tag{14}$$

Here S_i^{P2} is the new position determined by using the second phase of COA for the ith coati, $S_{i,j}^{P2}$ is the jth dimension, F_i^{P2} is the outcome of its objective function,

Step 4: Termination condition: Until the best hyper-parameter choice is achieved, the procedure is repeated. The selected hyperparameter value is applied to the improved vectorization. The COA pseudo-code is displayed in the table below.

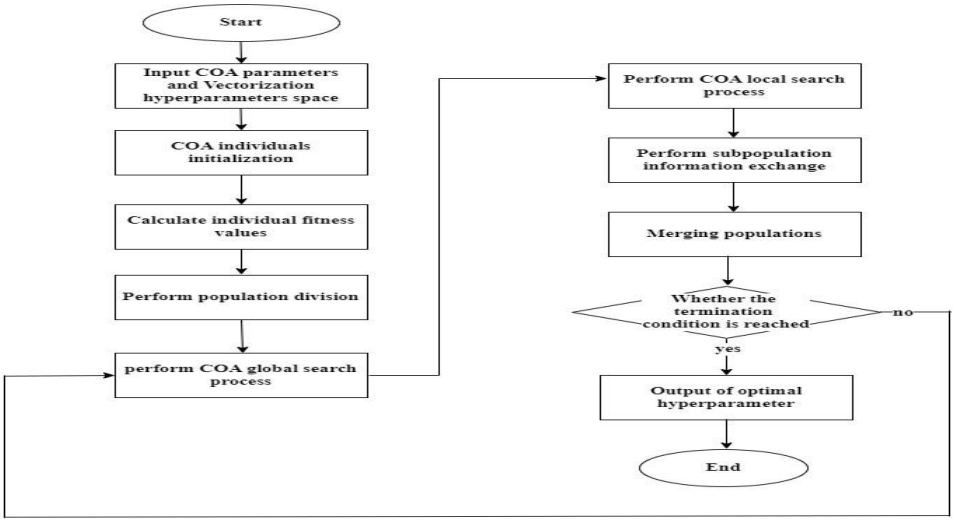

Fig. 3 Flowchart for COA

The proposed method utilizes a classification-based approach for text summarization, where each sentence is classified as either part of the summary or not. This classification is performed using a neural network, and to ensure optimal performance, the Coati Optimization Algorithm (COA) is employed for hyperparameter tuning. Specifically, COA optimizes key hyperparameters such as learning rate, batch size, dropout rate, and embedding dimension, which significantly impact the model's accuracy and efficiency. The optimization process begins with the initialization of a population of Coatis, where each represents a unique combination of hyperparameters. The fitness of each candidate solution is evaluated based on classification accuracy, and the positions of Coatis are updated using two strategies: the hunting and attacking strategy for global exploration and the escape from predators strategy for local refinement. This iterative process continues until the best set of hyperparameters is identified, which is then applied to the neural network model. As a result, while COA does not directly perform summarization, it plays a crucial role in enhancing the neural network's ability to accurately classify sentences, thereby improving the overall quality of the generated summary.

4. Result and Discussion

The findings show that the COA outperforms current optimization algorithms in producing succinct and insightful summaries, as evidenced by higher ROUGE and BLEU scores. The COA's potential as a potent tool for automatic text summaries is highlighted in the debate, which also highlights how well it extracts important information from documents while maintaining summary quality. A computer with an Intel (R) Core (TM) i5 4570s CPU running at 2.90 GHz, 8GB of RAM, Windows 64-bit, and Python was used for the experiments.

4.1. Dataset Description

In order to evaluate automated text summarizers, the dataset contains a variety of document collections as well as human-generated summaries. Each dataset contains single-document summaries with varying file sets (50, 30, and 50 sets, respectively) and differing numbers of files per set (12, 20, and 25). The type of documents ranges from human-written queries with summaries (DUC 2002), news articles with summaries (DUC 2003), to queries with five reference summaries (DUC 2005). These datasets are sourced from duc.nist.gov or TREC, with summaries containing average word counts of 112, 101, and 109, respectively.

4.2. Evaluation Metrics

It has chosen many metrics to gauge how well change-proneness prediction models are doing. They have selected such as ROUGE score, BLEU score, precision, recall, and F-score. The study compares the performance of the COA with existing state-of-the-art optimization algorithms such as PSO [29], CSO, GWO [30], Quick Artificial Bee colony optimization algorithm (QABC) [31], Modified cat swarm optimization algorithm (MCSO), Greedy local optimizer (GLO) [32].

Recall

Recall is a performance indicator that quantifies the percentage of pertinent information that is successfully extracted from the source text in machine learning tasks such as text summarization. It is computed as follows equation (15):

$$recall = \frac{TP}{TP + FN} \tag{15}$$

Precision

A performance parameter called precision is used to assess how accurately information is obtained in machine learning activities such as text summarization. It is computed as follows equation (16):

$$precision = \frac{TP}{TP + FP} \tag{16}$$

F1-score

A statistic called the F1-score is used to assess how well a summarization model balances recall and accuracy. The F1-score assigns equal weight to accuracy and recall by taking the harmonic mean of these two criteria. It is computed as follows equation (17):

$$F1 - Score = \frac{2TP}{2TP + FP + FN} \tag{17}$$

TP signifies the true positive, FP the false positive, TN the true negative, and FN the false negative.

ROUGE

ROUGE (Recall-Oriented Understudy for Gisting Evaluation) ratings were first introduced and have subsequently become widely recognized measures for assessing text summarization systems. The degree of overlap between machine-generated and human-written summaries is used to measure summarization quality in equation (18).

$$ROUGE = \sum_{v_j \in (v_i)} \frac{\sum S \in Refsummarizes \sum n - grams \in SCount_{match(n-gram)}}{\sum S \in Refsummarizes \sum n - grams \in S \ Count\left(n - gram\right)} \tag{18}$$

BLEU

The produced summary's word count is measured by BLEU (Bilingual Evaluation Understudy) in comparison to a reference summary in equation (19), (20),

$$BLUE = BP * \exp\left(\sum_{n-1}^{N} w_n \log p_n\right) \tag{19}$$

$$BP = \begin{cases} 1 \ \ if \ \ c > r \\ e^{1-\frac{r}{c}} \ \ if \ \ c \le r \end{cases} \tag{20}$$

Figure 4(a) presents ROUGE scores across three metrics—F-Measure, Precision, and Recall—highlighting performance for ROUGE-1, ROUGE-2, and ROUGE-L. In the chart, F-Measure is represented in blue, Recall in green, and Precision in red. The outcomes unequivocally show how effective the suggested method is in comparison to alternative strategies. The performance metrics of several methods on the single-document 2002 dataset are shown in Figure 4(b). The suggested method performs better than the others, especially when considering the F1-score of 0.51. All metrics show that the PSO and GWO techniques perform poorly. Overall, the proposed approach proves to be more effective than the other methods evaluated.

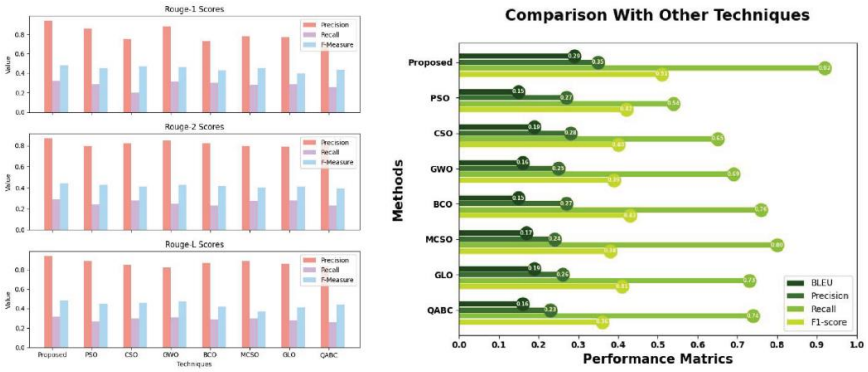

Fig. 4 Single-Document DUC 2002 dataset(a): Rouge-1, Rouge-2, Rouge-L results and (b) Evaluation metrics of proposed work with current techniques

ROUGE-1, ROUGE-2, and ROUGE-L scores for various techniques on the single-document 2003 dataset are shown in Figure 5(a).

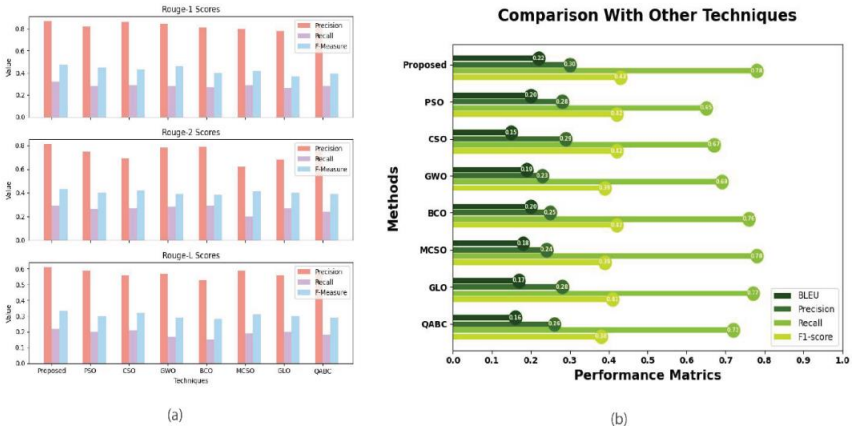

Fig. 5 Single-Document DUC 2003 dataset (a): comparison of proposed approach with existing methods using Rouge-1, Rouge-2, Rouge-L metrics (b): Analysis of Performance Metrics for Proposed work and Existing methods**.**

In all measures, the suggested technique excels, especially in ROUGE-1. The proposed methodology outperforms PSO and CSO in Recall and F-Measure, demonstrating its efficacy. Figure 5(b) compares Performance metrics like BLEU score, Precision, Recall, and F1-score for the same dataset. The proposed technique again beats alternatives with an F1-score of 0.78. In comparison, PSO and GWO lag in Precision and Recall.

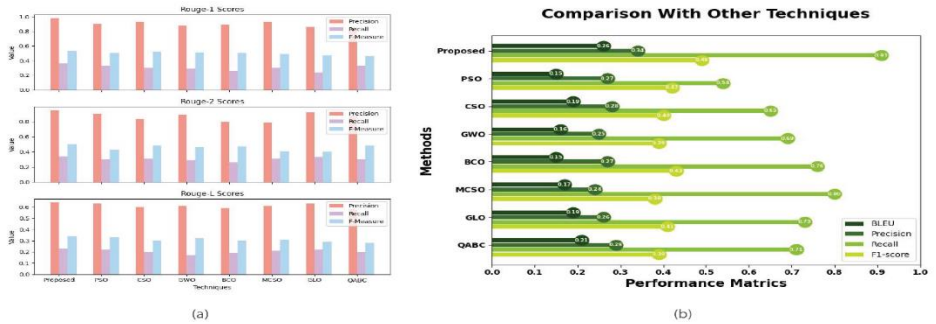

Fig. 6 Single-Document DUC 2005 dataset (a): Rouge-1, Rouge-2, Rouge-L scores between the proposed approach and existing methods. (b) Evaluation of performance metrics for the proposed method compared to existing approaches.

ROUGE-1, ROUGE-2, and ROUGE-L scores for different approaches on the single-document 2005 dataset are compared in Figure 6(a). The approach with the highest scores in all criteria excels in ROUGE-1 and ROUGE-L. The graph also shows Precision, Recall, and F-Measure, proving the technique works. The suggested algorithm outperforms PSO and CSO in Recall and F-Measure, proving its superiority. Figure 6(b) compares BLEU, Precision, Recall, and F1-score for the same dataset. All other methods fail to match the proposed method's 0.92 F1-score. PSO and GWO score lower in BLEU and Precision. Table 2 compares the suggested model's performance in detail.

Table 2 Comparative analysis of the proposed model

Dataset →	DUC 2002				DUC 2003				DUC 2005			
Techniques ↓	BLEU	Precision	Recall	F1 score	BLEU	Precision	Recall	F1 score	BLEU	Precision	Recall	F1 score
PSO	0.15	0.27	0.54	0.42	0.20	0.28	0.65	0.42	0.15	0.27	0.54	0.42
CSO	0.19	0.28	0.65	0.40	0.15	0.29	0.67	0.42	0.19	0.28	0.65	0.40
GWO	0.16	0.25	0.69	0.39	0.19	0.23	0.69	0.39	0.16	0.25	0.69	0.19
BCO	0.15	0.27	0.76	0.43	0.20	0.25	0.76	0.42	0.15	0.27	0.76	0.43
MCSO	0.17	0.24	0.80	0.38	0.18	0.24	0.78	0.39	0.17	0.24	0.80	0.38
GLO	0.19	0.26	0.73	0.41	0.17	0.28	0.77	0.41	0.19	0.26	0.73	0.41
QABC	0.16	0.23	0.74	0.36	0.16	0.26	0.72	0.38	0.21	0.29	0.71	0.39
Proposed	**0.29**	**0.35**	**0.92**	**0.51**	**0.22**	**0.30**	**0.78**	**0.41**	**0.26**	**0.34**	**0.91**	**0.49**

The proposed approach is evaluated against state-of-the-art optimization techniques, including PSO, CSO, GWO, BCO, MCSO, GLO, and QABC, using the DUC 2002, 2003, and 2005 datasets. Performance is assessed based on BLEU, precision, recall, and F1 score. The results indicate that our method surpasses existing techniques, particularly

in recall, achieving the highest values across all datasets—0.92 for DUC 2002, 0.78 for DUC 2003, and 0.91 for DUC 2005—demonstrating its effectiveness in preserving essential content. Additionally, it attains the highest BLEU score of 0.29 on DUC 2002, outperforming alternative methods, which range between 0.15 and 0.21. The F1 scores are also among the highest, peaking at 0.51 on DUC 2002, reflecting a well-balanced trade-off between precision and recall. However, while our approach excels in recall and F1 score, methods such as CSO and GLO achieve slightly comparable precision values, particularly in DUC 2003. This suggests that although our model retrieves a larger proportion of relevant sentences, further refinement may help reduce redundancy. Moreover, the computational complexity of COA warrants further investigation compared to other optimization techniques. Despite these considerations, the findings confirm that the proposed method significantly improves summarization performance, positioning it as a competitive alternative to existing state-of-the-art approaches.

5. Conclusion

The development of automated text summarization algorithms is crucial for efficiently extracting key information from large textual datasets, addressing the challenge of information overload in the digital era. This study introduces a systematic approach to single-document summarization by transforming words into vector representations and leveraging TF-ITF to assess sentence importance. The summarization process is further optimized using the Coati Optimization Algorithm (COA), which fine-tunes hyperparameters to enhance sentence ranking. Experimental results on benchmark datasets, including DUC 2002, 2003, and 2005, demonstrate that the COA-based approach outperforms state-of-the-art optimization techniques such as PSO, CSO, GWO, BCO, QABC, MCSO, and GLO, achieving higher recall and F-score values. By effectively refining sentence selection and improving vectorization, COA contributes to generating more informative and coherent summaries. The key contributions of this study include the integration of TF-ITF with COA for enhanced sentence ranking, a comprehensive comparative analysis with multiple optimization techniques, and the optimization of hyperparameters to improve summarization performance. Future work will focus on extending this approach to multi-document summarization and exploring deep learning-based hybrid models to further enhance summary quality.

References

[1] W. Kryściński, N. S. Keskar, B. McCann, C. Xiong and R. Socher, "Neural Text Summarization: A Critical Evaluation", *arXiv preprint*, arXiv:1908.08960, 2019.

[2] J. Weston, "A Neural Attention Model for Abstractive Sentence Summarization", *arXiv preprint*, arXiv:1509.00685, 2015.

[3] L. Liu, Y. Lu, M. Yang, Q. Qu, J. Zhu and H. Li, "Generative Adversarial Network for Abstractive Text Summarization", In Proceedings of the AAAI Conference on Artificial Intelligence, vol. 32, no. 1, Apr. 2018, pp. 1-3.

[4] D. Miller, "Leveraging BERT for Extractive Text Summarization on Lectures", *arXiv preprint*, arXiv:1906.04165, 2019.

[5] K. Sarkar, "Automatic Single Document Text Summarization Using Key Concepts in Documents", *J. Inf. Process. Syst.*, vol. 9, no. 4, pp. 602-620, 2013.

[6] H. Christian, M. P. Agus and D. Suhartono, "Single Document Automatic Text Summarization Using Term Frequency-Inverse Document Frequency (TF-IDF)", *ComTech: Comput. Math. Eng. Appl.*, vol. 7, no. 4, pp. 285-294, 2016.
[7] R. Z. Al-Abdallah and A. T. Al-Taani, "Arabic Single-Document Text Summarization Using Particle Swarm Optimization Algorithm", *Procedia Comput. Sci.*, vol. 117, pp. 30-37, 2017.
[8] U. Rani and K. Bidhan, "Review Paper on Automatic Text Summarization", *Int. Res. J. Eng. Technol.* (IRJET), vol. 7, no. 4, pp. 3349-3354, 2020.
[9] A. A. Syed, F. L. Gaol and T. Matsuo, "A Survey of the State-Of-The-Art Models in Neural Abstractive Text Summarization", *IEEE Access*, vol. 9, pp. 13248-13265, 2021.
[10] S. Sivakumar and R. Rajalakshmi, "Context-aware Sentiment Analysis with Attention-Enhanced Features from Bidirectional Transformers", *Soc. Netw. Anal. Min.*, vol. 12, no. 1, p. 104, 2022.
[11] R. Srivastava, P. Singh, K. P. S. Rana and V. Kumar, "A Topic Modeled Unsupervised Approach to Single Document Extractive Text Summarization", *Knowl.-Based Syst.*, vol. 246, p. 108636, 2022.
[12] P. Verma, A. Verma and S. Pal, "An Approach for Extractive Text Summarization Using Fuzzy Evolutionary and Clustering Algorithms", *Appl. Soft Comput.*, vol. 120, p. 108670, 2022.
[13] D. V. P. Kumar, S. S. Raj, P. Verma and S. Pal, "Extractive Text Summarization Using Meta-Heuristic Approach", in *FIRE (Working Notes)*, pp. 464-474, 2022.
[14] J. Cheng and M. Lapata, "Neural Summarization by Extracting Sentences and Words", *arXiv preprint*, arXiv:1603.07252, 2016.
[15] W. Kryściński, B. McCann, C. Xiong and R. Socher, "Evaluating the Factual Consistency of Abstractive Text Summarization", *arXiv preprint*, arXiv:1910.12840, 2019.
[16] D. Debnath, R. Das and P. Pakray, "Extractive Single Document Summarization Using an Archive-Based Micro Genetic-2", In Proceedings of the 7th International Conference on Soft Computing & Machine Intelligence (ISCMI), 2020, pp. 244-248.
[17] T. Bezdan et al., "Hybrid Fruit-Fly Optimization Algorithm with k-Means for Text Document Clustering", *Mathematics*, vol. 9, no. 16, p. 1929, 2021.
[18] D. Debnath, R. Das and P. Pakray, "Single Document Text Summarization Addressed with A Cat Swarm Optimization Approach", *Appl. Intell.*, vol. 53, no. 10, pp. 12268-12287, 2023.
[19] S. P. Patil and R. Rautray, "SMATS: Single and Multi Automatic Text Summarization", *Karbala Int. J. Modern Sci.*, vol. 9, no. 1, p. 6, 2023.
[20] K. Svore, L. Vanderwende and C. Burges, "Enhancing Single-Document Summarization by Combining Ranknet and Third-Party Sources", In Proceedings of the Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning (EMNLP-CoNLL), 2007, pp. 448-457.
[21] S. Mandal, G. K. Singh and A. Pal, "Single Document Text Summarization Technique Using Optimal Combination of Cuckoo Search Algorithm, Sentence Scoring and Sentiment Score", *Int. J. Inf. Technol.*, vol. 13, no. 5, pp. 1805-1813, 2021.
[22] A. Jain, D. Yadav and A. Arora, "Particle Swarm Optimization for Punjabi Text Summarization", *Int. J. Oper. Res. Inf. Syst.* (IJORIS), vol. 12, no. 3, pp. 1-17, 2021.
[23] S. H. Apandi, J. Sallim, R. Mohamed and N. Ahmad, "Data Pre-Processing of Website Browsing Records: To Prepare Quality Dataset for Web Page Classification", *JOIV: Int. J. Inf. Visual.*, vol. 8, no. 1, pp. 239-246, 2024.
[24] M. Jaiswal and S. Das, "Detecting Spam E-Mails Using Stop Word TF-IDF and Stemming Algorithm with Naïve Bayes Classifier on the Multicore GPU", *Int. J. Electr. Comput. Eng.*, vol. 11, no. 4, pp. 3168-3175, 2021.
[25] K. K. Mohbey and S. Tiwari, "Preprocessing and Morphological Analysis in Text Mining", *Int. J. Electron. Commun. Comput. Eng.*, vol. 2, no. 2, pp. 116-122, 2011.
[26] Z. Gou, Z. Huo, Y. Liu and Y. Yang, "A Method for Constructing Supervised Topic Model Based on Term Frequency-Inverse Topic Frequency", *Symmetry*, vol. 11, no. 12, p. 1486, 2019.
[27] A. K. Singh and M. Shashi, "Vectorization of Text Documents for Identifying Unifiable News Articles", *Int. J. Adv. Comput. Sci. Appl.*, vol. 10, no. 7, pp. 305-310, 2019.
[28] M. Dehghani, Z. Montazeri, E. Trojovská and P. Trojovský, "Coati Optimization Algorithm: A New Bio-Inspired Metaheuristic Algorithm for Solving Optimization Problems", *Knowl.-Based Syst.*, vol. 259, p. 110011, 2023.
[29] V. Dalal and L. Malik, "Semantic Graph-Based Automatic Text Summarization for Hindi Documents Using Particle Swarm Optimization", in *Information and Communication Technology for Intelligent Systems*, Springer, pp. 284-289, 2018.

[30] N. Saini, S. Saha, A. Jangra and P. Bhattacharyya, "Extractive Single Document Summarization Using Multi-Objective Optimization: Exploring Self-Organized Differential Evolution, Grey Wolf Optimizer and Water Cycle Algorithm", *Knowl.-Based Syst.*, vol. 164, pp. 45-67, 2019.
[31] J. Rautaray et al., "SEQABC: Revolutionizing Single Document Extractive Text Summarization with Quick Artificial Bee Colony", *Nanotechnol. Percept.*, pp. 737-750, 2024.
[32] M. Mendoza, C. Cobos and E. León, "Extractive Single-Document Summarization Based on Global-Best Harmony Search and A Greedy Local Optimizer", in *Advances in Artificial Intelligence and Its Applications*, Springer, pp. 52-66, 2015.
[33] H. Zhang, P. S. Yu and J. Zhang, "A Systematic Survey of Text Summarization: From Statistical Methods to Large Language Models", *arXiv preprint*, arXiv:2406.11289, 2024.
[34] D. Yadav, J. Desai and A. K. Yadav, "Automatic Text Summarization Methods: A Comprehensive Review", arXiv preprint, arXiv:2204.01849, Mar. 2022.
[35] S. Mirjalili, S. M. Mirjalili and A. Lewis, "Grey Wolf Optimizer", *Adv. Eng. Soft.*, vol. 69, pp. 46-61, 2014.
[36] D. Karaboga and B. Basturk, "A Powerful and Efficient Algorithm for Numerical Function Optimization: Artificial Bee Colony (ABC) Algorithm", *J. Glob. Optim.*, vol. 39, no. 3, pp. 459-471, 2007.
[37] G. Wang, S. Deb and L. Zhao, "A Multi-Colony Multi-Objective Particle Swarm Optimizer for Dynamic Optimization Problems", *Eng. Appl. Artif. Intell.*, vol. 62, pp. 3-15, 2017.
[38] S. Sharma, A. Verma and P. K. Shukla, "A Novel Glowworm Swarm Optimization Algorithm for Text Document Summarization", *Expert Syst. Appl.*, vol. 160, p. 113653, 2020.
[39] X. Yuan, Y. Xu, L. Gao and Y. Zhang, "A Quick Artificial Bee Colony Algorithm for Large-Scale Numerical Optimization", *Appl. Soft Comput.*, vol. 48, pp. 579-596, 2016.