AIMS Mathematics

AIMS Mathematics, 9(5): 12588–12609.
DOI: 10.3934/math.2024616
Received: 21 January 2024
Revised: 29 February 2024
Accepted: 06 March 2024
Published: 01 April 2024

https://www.aimspress.com/journal/Math

Research article

Enhancing extractive text summarization using natural language processing with an optimal deep learning model

Abdulkhaleq Q. A. Hassan[1], Badriyya B. Al-onazi[2], Mashael Maashi[3], Abdulbasit A. Darem[4,*], Ibrahim Abunadi[5] and Ahmed Mahmud[6]

1 Department of English, College of Science and Arts at Mahayil, King Khalid University, Saudi Arabia
2 Department of Language Preparation, Arabic Language Teaching Institute, Princess Nourah bint Abdulrahman University, P.O. Box 84428, Riyadh 11671, Saudi Arabia
3 Department of Software Engineering, College of Computer and Information Sciences, King Saud University, P.O. Box 103786, Riyadh 11543, Saudi Arabia
4 Department of Computer Science at the College of Science, Northern Border University, Arar, Saudi Arabi
5 Department of Information Systems, College of Computer and Information Sciences, Prince Sultan University, Riyadh, Kingdom of Saudi Arabia
6 Research Center, Future University in Egypt, New Cairo 11835, Egypt

*** Corresponding:** Email: basit.darem@nbu.edu.sa.

Abstract: Natural language processing (NLP) performs a vital function in text summarization, a task targeted at refining the crucial information from the massive quantity of textual data. NLP methods allow computers to comprehend and process human language, permitting the development of advanced summarization methods. Text summarization includes the automatic generation of a concise and coherent summary of a specified document or collection of documents. Extracting significant insights from text data is crucial as it provides advanced solutions to end-users and business organizations. Automatic text summarization (ATS) computerizes text summarization by decreasing the initial size of the text without the loss of main data features. Deep learning (DL) approaches exhibited significant performance in abstractive and extractive summarization tasks. This research designed an extractive text summarization using NLP with an optimal DL (ETS-NLPODL) model. The major goal of the ETS-NLPODL technique was to exploit feature selection with a hyperparameter-tuned DL model for summarizing the text. In the ETS-NLPODL technique, an initial step of data preprocessing was involved to convert the input text into a compatible format. Next, a feature extraction process was carried out and the optimal set of features was chosen by the hunger games search optimization (HGSO) algorithm. For text summarization, the ETS-NLPODL model used an attention-based convolutional neural network with a gated recurrent unit (ACNN-GRU) model. Finally, the mountain gazelle optimization (MGO) algorithm was employed for the optimal hyperparameter selection of the ACNN-GRU model. The experimental results of the ETS-NLPODL system were examined under the benchmark dataset. The experimentation outcomes pointed out that the ETS-NLPODL technique gained better performance over other methods concerning diverse performance measures.

Keywords: text summarization; natural language processing; mountain gazelle optimization; feature extraction; deep learning
Mathematics Subject Classification: 11Y40

1. Introduction

Text summarization is a difficult task in NLP. Its goals are to perform highly controllable reading and search data from numerous documents by making minor versions without losing importance [1]. Due to the Internet's rapid development during the last couple of decades, book reviews, articles, and data availability news have been introduced on the Internet that can be improved quickly [2]. There is a substantial improvement in textual information and it is constantly growing because of the enormous data quantity. Users employ search queries to find information on the Internet. In addition, the user should search several web pages, which takes time and can cause a headache, to determine the accuracy of the data that they need [3]. Therefore, to evade this headache, handle this large quantity of data, and acquire the data from whole documents in a short time, a technique has been presented called text summarization. According to the kind of summary that must be created, text summarization can be categorized into two types: extractive and abstractive text summarization (ATS). Extractive summarization depends mostly on linguistic or statistical factors to extract the main parts of the source text verbatim [4]. Although the abstractive summarization repeats the acquired text for producing words, which could not involved in the source text in incompatible with repeating any parts of the actual text. Making the summary employing NLP and advanced machine learning (ML) methods produces ATS at a higher complexity than extractive text summarization [5]. These resources should be analyzed for the use of offering abstract summaries semantically.

Recently, numerous developments have arisen in extractive summarization from simple approaches (e.g., topical-based, statistical-based, graph-based, etc.) as well as further advanced techniques (ML-based and DL-based) [6]. In ML, the summarization issue has changed into a supervised classification issue at the level of sentences. The technique finds that if a sentence in a test article is a portion of a summary, analysis will depend on evaluating a set of documents as well as their summaries in a training phase [7]. Summarization methods dependent upon ML begin with recognizing features from preprocessed forms like sentence length, sentence location, proper nouns, sentence importance, etc., followed by sorting those features using ML classification methods to produce a particular score as an output [8]. However, manual engineering has changed recently to data-driven approaches employing DL. Consequently, a neural network (NN) with appropriate depth could be implemented for extracting features and classifying sentences as important to exclude or include in the summary. As NN takes only numbers as input, however, the text has a string, and word embeddings have been employed to overcome this problem [9]. In this approach, words should be converted into vectors in a proper space called word vectors. By employing word vectors, linguistic predictabilities and patterns have been converted clearly, inspiring NNs to identify linguistic features explicitly [10]. Because of this notion, DL-based methods have achieved popularity in extractive summarization.

This research designed an extractive text summarization using NLP with an optimal DL (ETS-NLPODL) model. The major goal of the ETS-NLPODL technique was to exploit feature selection with a hyperparameter-tuned DL method for summarizing the text. In the ETS-NLPODL algorithm, the initial step of data preprocessing was involved to convert the input text into a compatible format. Next, a feature extraction process was carried out and the optimal set of features was chosen by the hunger games search optimization (HGSO) algorithm. For text summarization, the ETS-NLPODL model used an attention-based convolutional NN with a gated recurrent unit (ACNN-GRU) model. Finally, the mountain gazelle optimization (MGO) algorithm was applied for the optimal hyperparameter selection of the ACNN-GRU model. The experimentation results of the ETS-NLPODL technique were then tested on a benchmark dataset.

2. Related works

Babu and Badugu [11] considered the ATS technique for the Telugu language for making a clear summary. The popular analysis of the ATS method was carried out in English, whereas no important investigation in Telugu could be recognized. An abstractive Telugu text summarization method dependent upon a sequence-to-sequence (seq2seq) encoder-decoder framework was developed. The seq2seq architecture was applied with an LSTM-based decoder and bi-directional-LSTM (Bi-LSTM)-based encoder. Tripathy and Sharmila [12] deal with the issue of interpretation of this wide-ranging data by summarizing it employing a text summarizer dependent on ATS via DL techniques like employing the pointer generator mode and Bi-LSTM networks. These study goals offer an analysis of these architectures to achieve good knowledge of the model’s performance in order to create a robust text summarizer.

Shafiq et al. [13] designed a DL method for the Urdu language by employing the Urdu one million information databases as well as correlating their effectiveness with two extensively employed methods depending on ML, namely logistic regression (LR) and support vector machines (SVM). The outcomes demonstrated that the developed DL techniques worked better than the other two methods. The summary created by extractive summarization was processed through the encoder-decoder model for producing an abstractive summarization. In [14], the authors attempted to solve the database scarcity in Indian Regional Languages, such as Marathi and Hindi, as well as develop two novel DL approaches to execute text summarization employing the abstractive method that includes attention and stacked LSTM-based Seq2Seq-NN.

Rani and Lobiyal [15] projected an ATS architecture dependent on the document vector technique for analyzing semantic relationships present in an article. Additionally, the authors offered two methods, sentence ranking and summary production, dependent on three major features comprising compression rate, diversity, and redundancy for making perfect and coherent summaries. The developed method was language-independent with any particular language preprocessing. Etaiwi and Awajan [16] introduced an innovative semantic graph embedding-based ATS approach for the Arabic language like SemG-TS. It utilized deep NNs (DNNs) for producing the abstractive summarization. A collection of tests have been performed for analyzing the effectiveness of SemG-TS, as well as comparing the outcomes to a prevalent baseline word-embedding method named word2vec.

AlTimimi and AlRubbiay [17] presented an innovative technique for multi-document summarization. The simple concepts of this developed technique depend on various features that are removed from each sentence, these features should be language. The feature vector sets were presented to CNNs for classifying as both non-summary and summary sentences. A summary sentences graph was produced and allocated scores by the TextRank method. Moravvej et al. [18] provided a supervised extractive summarization technique dependent upon modified generative adversarial networks (GANs) employing CNNs. Different from earlier methods that repeatedly utilized greedy approaches for choosing sentences, the authors employed a novel technique to choose sentences. In addition, the authors offered a network for biomedical word-embedding that increased summarization.

3. The proposed method

In this manuscript, we focus on the designs and development of the ETS-NLPODL algorithm. The major goal of the ETS-NLPODL technique is to exploit feature selection with a hyperparameter-tuned DL model for summarizing the text. Figure 1 depicts the entire flow of the ETS-NLPODL algorithm.

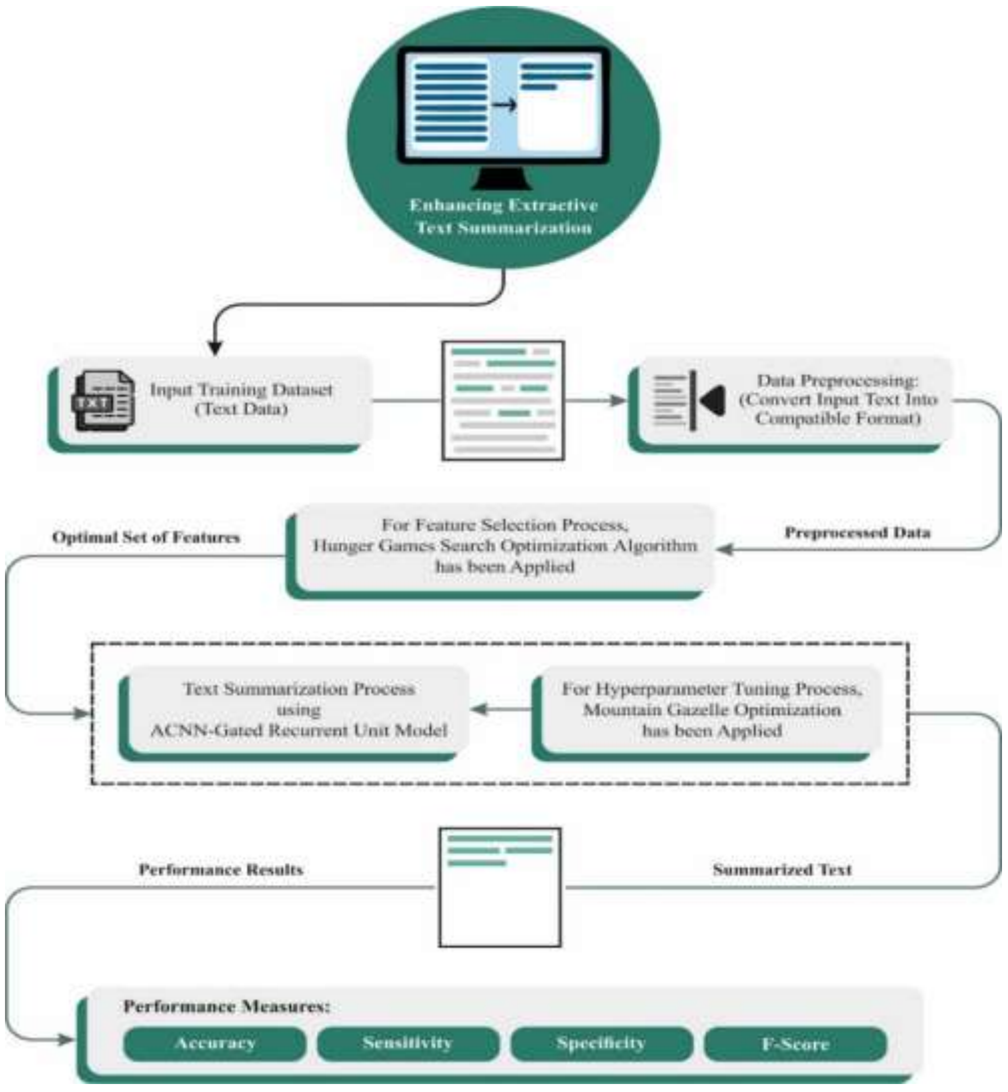
Enhancing Extractive Text Summarization
Input Training Dataset (Text Data)
Data Preprocessing: (Convert Input Text Into Compatible Format)
Preprocessed Data
For Feature Selection Process, Hunger Games Search Optimization Algorithm has been Applied
Optimal Set of Features
Text Summarization Process using ACNN-Gated Recurrent Unit Model
For Hyperparameter Tuning Process, Mountain Gazelle Optimization has been Applied
Summarized Text
Performance Results
Performance Measures:
Accuracy
Sensitivity
Specificity
F-Score

Figure 1. Overall flow of the ETS-NLPODL algorithm.

3.1. Preprocessing

Initially, data preprocessing is involved to convert the input text into a compatible format. Removing the data and getting rid of some uncertainties from the database are the primary objectives of preprocessing [19]. The presented method preparation mostly requires nine main steps: empty record removal, link removal, stop words removal, lower space removal, stemming, lemmatization removal, visualization removal, and data limitation removal. The database has been preprocessed employing these unique procedures to prepare it for short text summarization.

Link removing: The preprocessing of data also comprises eliminating redundant links to obtain quality solutions. The links could not involve shorter text but summarisation and are regarded as trash. There are various procedures utilized for removing the links in the text data. other than Natural Language Tool Kit (NLTK) so it can be slow from the speed level fixed expression carried out gradually in difficult functions.

Removing of empty records: During the preprocessing stage, the whole database has been examined and the empty records from the database are eliminated. The elimination of the empty records is executed to obtain the data from the method that processed more processes. This method offers to removal of the database and will support making a shorter text summarization.

Lowercasing: It can be vital before processing the text; our presented method executes lowercase. This will allow similar words to be examined, even in the case of two words having an identical meaning. During the preprocessing step but generating the text summarization most significant, the lowercase was further required.

Stop word removal: The necessity for stop word removal focuses on preprocessing and receiving the correct solutions. In this stage, it can be initially turned into words in the textual data followed by word-by-word cleaning. Stop words can be removed utilizing group frequency that records every occurrence of a word from the document.

Stemming: The stemming model was applied from the preprocessing step, an essential method for performing the short text summarization. This method proceeds word as well as changes into the root word.

Lemmatization: Lemmatization is performed to convert the words in varied procedures of the words near the simple procedure of the word for easing the analyzation model and preprocessing step. Mostly, the drive of the lemmatization can improve the reliability of the model by creating words very apparent.

Data limitation: Data limitation starts with the text length once a limit is carried out, to create a text, which has a brief sufficient to adapt the provided data. This data limitation has been employed to achieve specific short text-length data. These data limitations are specially utilized for this model to entertain a certain data length that is defined as a brief sentence, for maintaining the importance of the objective.

3.2. Feature extraction

At this stage, the highly significant sentences have been carefully chosen in the group of features [20]. The input document has been preprocessed as a vector of features. This attribute is used to represent summary sentences. Fifteen features are extracted. Each feature provides different values, while the low value highlights a strong presence of features and the high value indicates a low presence of features in the sentence. The group of text and static features used for the presented method is discussed below:

Term frequency (F_1): Term frequency (F) indicates the total time a word is located inside the article. When the word frequency within the text is high, then the word has a significant effect on the document text.

Sentence length (F_2): Each sentence specifies a score dependent on length, explicitly the number of words existing in certain sentences.

Sentence position value (F_3): The sentence positional value in the text defines the document's importance. The initial sentence indicates the theme of the document, whereas the ending sentence infers or wraps up the document. The value can be measured by allocating a high score value to the initial and the ending sentence of the document.

Keywords (F_4): This is an important word in the document that is provided as input. When the sentence includes keywords, allocate a score value of "1"; otherwise, the value is "0."

Similarity with title (F): This is described by the count of words to be intersection amongst title and query as well as defined if the title signifies a similar entity or not. This includes the words in headers and titles. Also, they have a few extra weights in sentence scores.

Proper noun (F_6): If a proper noun exists in the sentence, then more weights are specified in the sentence. The sentences with proper nouns must be considered as significant.

Nouns and verbs (F_7): A sentence that has several nouns and verbs is important data.

Sentence similarity (F_8): Sentence similarity (F) describes the vocabulary overlap between the two sentences.

Font style (F_9): This provides higher weight to uppercase words and lower weight to lowercase letters.

Word similarity among sentences (F_{10}): Word similarity among sentences (F) defines what kind of close the two words have been linked.

Word similarity among paragraphs (F_{17}): If a given word in a paragraph occurs recurrently in other sentences, the next paragraph sentence is considered more significant.

Cue word (F_{12}): A cue word has a connective expression, which connects the semantic relationship and span of discourse in the texts. As they are frequently given as input, then it is a significant word in the document. If the sentence has a cue word, the score value is allocated to the sentence; otherwise, it is given a score of zero.

Positive and negative keywords (F_{13}): The negative keyword less frequently occurs in the summaries and should have a certain negative weight. The positive keyword commonly occurs in the summary and should be specified a high weight equal to the cue word.

Numerical word (F_{14}): Sentences containing arithmetical information are the significant ones for the summary, hence weight must be allocated for the arithmetical information.

3.3. HGSO-based feature selection

At this stage, the HGSO model is executed for an optimum set of features. Yang et al. introduced an HGS algorithm in 2021 motivated by behavioral choices and hunger-driven activities among the animal's instinct to survive [21]. Social animals mainly take part in foraging, but some individuals may not cooperate. Eq (1) provides the foraging behavior and cooperative communication of game rules:

$$\overrightarrow{X(t+1)} = \begin{cases} \text{Game}_1 : \overrightarrow{X(t)} \cdot \left(1 + randn(1)\right), r_1 < l \\ \text{Game}_2 : \overrightarrow{W_1} \cdot \rightarrow X_b + \vec{R} \cdot \overrightarrow{W_2} \cdot \left|\overrightarrow{X_b} - \overrightarrow{X(t)}\right|, r_1 > l, r_2 > E \\ \text{Game}_3 : \overrightarrow{W_1} \cdot \rightarrow \chi_b - \vec{R} \cdot \overrightarrow{\mathrm{W}_2} \cdot \left|\overrightarrow{X_b} - \overrightarrow{X(t)}\right|, r_1 > l, r_2 < E \end{cases} \tag{1}$$

where r_1 and $r_{2/}$ are the random integers ranging from zero and one, and $randn(1)$ denotes the uniformly distributed random value.

W_1 and W_2 are the hunger weights. $X(t)$ and X_b are the location of every agent and the optimal agent, correspondingly. The parameter l assists in improving the algorithm. R designates the number in $[-\alpha, \alpha]$ that is evaluated by the following equation:

$$\vec{R} = 2 \times shrink \times rand - shrink \tag{2}$$

In Eq (2), $shrink = 2 \times \left(1 - \frac{t}{T}\right)$, $rand$ is a random integer in [0,1] and the maximal iteration count is T. The E parameter is a variation control for each position as follows:

$$E = sech(|F(i) - BF|), i \in 1,2, \cdots, n \tag{3}$$

In Eq (3), $F(i)$ and BF are the fitness values of all the agents and the optimum fitness value, respectively. The hyperbolic function of $sech$ is represented by $2/(e^x + e^{-x})$. The starvation features of the individual are formulated by the hunger weight that is shown below:

$$\overrightarrow{W_1(\iota)} = \begin{cases} hng(i) \cdot \frac{N}{Shng} \times r_4, r_3 < l \\ 1, r_3 > l \end{cases} \tag{4}$$

$$\overrightarrow{\mathrm{W}_2(\iota)} = \left(1 - e^{-|hng(i) - Shng|}\right) \times r_5 \times 2 \tag{5}$$

In Eqs (4) and (5), hng indicates the hunger for individuals and the number of individuals is N. In addition, r_3, r_4, and r_5 are the random integers between [0,1], and hng indicates the addition of hungry feelings of every individual, which can be given by:

$$hng(i) = \begin{cases} 0, Allfit(i) == BF \\ hng(i) + H, Allfit(i)! = BF \end{cases} \tag{6}$$

In Eq (6), BF indicates the optimum fitness value and $Allfit(i)$ denotes the fitness of all the agents in the existing iteration.

The H indicates the hunger sensation that can be evaluated by the following expression:

$$H = \begin{cases} LH \times (1 + r), TH < LH \\ TH, TH \geq LH \end{cases} \tag{7}$$

where $TH = (F(i) - BF/(WF - BF)) \times r_6 \times 2 \times (UB - LB)$. WF shows the worst fitness value, LH refers to the lower bound, and UB and LB denote the feature space-relevant lower and upper boundaries. $F(i)$ denotes the fitness value of all the agents.

In this method, the objectives must be combined into a single objective equation like a preset weight that recognizes every objective position [22]. It accepts a FF that integrates both objectives of FS as revealed in (8).

$$Fitness(X) = \alpha \cdot E(X) + \beta * \left(1 - \frac{|R|}{|N|}\right) \tag{8}$$

where $E(X)$ denotes the classifier error rate by employing the chosen features, Fitness (X) signifies the fitness value of a subset X, $|R|$ and $|N|$ are the number of chosen features as well as the number of actual features in the dataset, respectively, and α and β represent the weights of the classifier error and reduction ratio, $\alpha \in [0,1]$ and $\beta = (1 - \alpha)$.

3.4. Classification using ACNN-GRU model

For text summarization, the ETS-NLPODL model uses the ACNN-GRU model. The CNN-GRU module involves a 1D-CNN and 2 GRU layers, correspondingly [23]. The resultant of the CNN-GRU network is evaluated as:

$$\begin{gathered} C_t = \sigma(W_s * X_t + b_s), \\ z_t = \sigma(W_z i_t + U_z h_{t-1}), \\ r_t = \sigma(W_r i_t + U_r h_{t-1}), \\ \tilde{h}_t = \tanh(W_h i_t + r_t \circ U_h h_{t-1} + b_h), \\ h_t = (1 - z_t) \circ h_{t-1} + z_t \circ \tilde{h}_t, \end{gathered} \tag{9}$$

where σ stands for the sigmoid activate the function, X_t refers to the RFSs calculated above, i_t implies the resultant of the CNN layer, W and U define the weight and parameter matrix of the GRU unit, $*$ represents the convolutional function, W_s and b_s denote weighted and biased of the CNN algorithm, z_t and r_t signify the resultant of update and reset gates at the earlier moment t, $\circ$ signifies the Hadamard product, h_t demonstrates the layer at time t, and $\tilde{h}_t$ refers to the candidate layer.

Attention models are planned to remove the intrinsic features of instances as well as improve the speed and solution of feature processes. To address the limitations of the RNN technique in emphasizing vital data, the self-attention (SA) procedure has been employed to concentrate on the decreased features to be efficient for the summarization model. Figure 2 represents the structure of the ACNN-GRU model.

Once we have attained the attention weight of every time step, the last outcome H_t^o of the attention-based CNN-GRU is computed as a weighted sum of the resultant CNN-GRU h_t given by:

$$H_t^o = \sum_{k=1}^{n+1} \alpha_k \, h_{t-n+1}, \tag{10}$$

where $n + 1$ refers to the flow size and α_k represents the attention value with time $t - (k - 1)$. α_k can be written as:

$$\alpha_k = \frac{\exp(e_k)}{\sum_{k=1}^{m+1} \exp(e_k)}. \tag{11}$$

The scores $e = (e_1, e_2, e_3 \; e_{n+1})$ indicate the significance of all the positions from the data, and they have been measured as:

$$e_t = v_s \tanh(W_s C_t + U_s h_t), \tag{12}$$

in which e_t denotes the score of every time, and v_s, W_s, and U_s represent the hyperparameters that are learned.

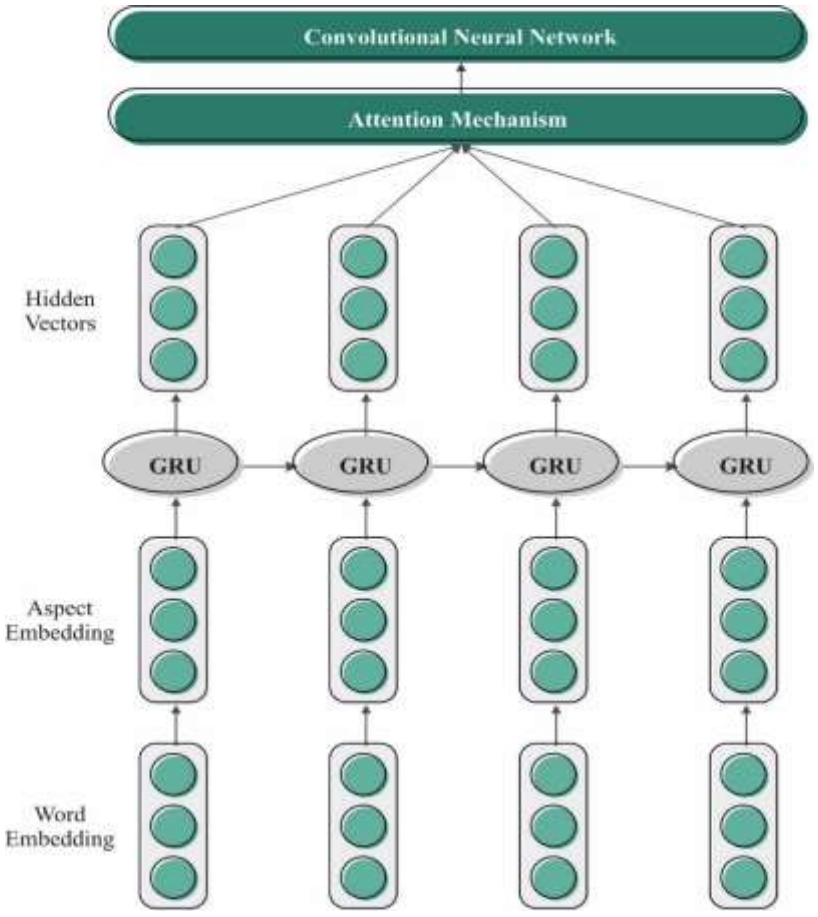

Figure 2. Architecture of the ACNN-GRU model.

3.5. MGO-based hyperparameter tuning

Eventually, the MGO model can be implemented for the optimum hyperparameter selection of the ACNN-GRU model. MGO is a recently established metaheuristic algorithm inspired by the natural group and social behaviors of mountain deer [24]. Territorial solitary males, maternity herds, bachelor male herds, and relocation looking for food can be the four primary aspects of mountain gazelle life where the MGO technique achieves optimization operation. The strategy used by ṀGO to implement optimization operation and the mathematical modeling can be given by the following:

3.5.1. Territory solitary males (TSM)

As soon as the male gazelles become adults, they are physically capable of defending themselves and establishing highly protected and solitary territories that can be divided by distance. Adult males conflict with one another for controlling the females' territory or ownership. Mature male gazelles attempt to defend the environments, whereas the young males attempt to the territory:

$$TSM = male_{gazelle} - |\beta_1 * BH - \beta_2 * X(t) * F| * Cof_r \tag{13}$$

$$BH = X_{ra} * \lfloor r_1 \rfloor + M_{pr} * \lceil r_2 \rceil, ra = \left\{\left\lceil \frac{N}{3} \right\rceil \dots N\right\} \tag{14}$$

$$F = N_1(D) * exp\left(2 - it * \left(\frac{2}{\text{Max } it}\right)\right) \tag{15}$$

$$Cof_r = \begin{cases} (a+1) + r_3, \\ a * N_2(D), \\ r_4(D), \\ N_3(D) * N_4(D)^2 * cos\big(2r_4 N_3(D)\big) \end{cases} \tag{16}$$

$$a = -1 + it * \frac{-1}{\text{Max } it} \tag{17}$$

Now, the location vector of a best male adult is represented as $male_{gazelle}$. The initial location of the gazelle is $X(t)$. β_1, β_2, β_3, and β_4 are random integers within [1,2]. The coefficient vectors of adolescent males are recognized as BH. Also, the random coefficient vector Cof_r has been updated after each round to increase the efficiency of the search area. The adolescent male in the range of ra signifies X_{ra}, and the average number of random population $\left\lceil \frac{N}{3} \right\rceil$ indicates M_{pr}. The entire population of gazelles is N, and the random integers within [0,1] are r_1, r_2, r_3, and r_4. N_1 shows the uniformly distributed random value. The random integers and dimensions of the problem are N_2, N_3, and N_4. cos denotes the cosine function and exp is the exponential function. Lastly, $Maxit$ and it are the maximal and existing number of generations, respectively.

3.5.2. Maternity herds (MH)

The gazelle is a type of animal that is dependent upon maternity herds to generate strong male offspring, which is an essential measure of their lifecycle. Also, male gazelles help with gazelle birth and adolescent males who are trying to possess female gazelles:

$$\text{MH} = \left(BH + Cof_{1,r}\right) + \left(\beta_3 * male_{gazelle} - \beta_4 * X_{rand}\right) * Cof_{2,r} \tag{18}$$

A gazelle is selected randomly from the overall population, and its vector location is characterized as X_{rand}. The coefficient vectors $Cof_{1,r}$ and $Cof_{2,r}$ are selected at random.

3.5.3. Bachelor male herd (BMH)

Male gazelles tend to generate territories and take control over female gazelles once they mature. In addition, adolescent males start to fight with adults for supremacy over females, and violence will occur, which can be mathematically modeled below:

$$BMH = (X(t) - D) + \left(\beta_5 * male_{gazelle} - \beta_6 * BH\right) * Cof_r \tag{19}$$

$$D = \left(|X(t)| + \left|male_{gazelle}\right|\right) * (2 * r_6 - 1) \tag{20}$$

Now, the vector place of the gazelle in the existing round has been represented as $X(t)$. β_5 and β_6 are the random integers within [1,2]. r_6 is the random value ranging from zero to one.

The MGO model derives an FF to obtain increased efficiency of classification. It finds a positive integer to denote the greater performance of the candidate solutions. A decrease in classifier error rate is measured as the FF and is described in Eq (21).

$$fitness(x_i) = ClassifierErrorRate(x_i)$$

$$= \frac{No.\ of\ misclassified\ Instances}{Total\ no.\ of\ Instances} * 100 \tag{21}$$

4. Result analysis

This section tests the performance of the ETS-NLPODL system concerning various evaluations. Table 1 investigates a detailed result analysis of the ETS-NLPODL technique on a single document [25]. Figure 3 offers a $sens_y$ outcome of the ETS-NLPODL system with other algorithms. The achieved outcome exhibits that the ETS-NLPODL system attained boosted outcomes with each file size. According to the 1000kb file, the ETS-NLPODL model offered a better $sens_y$ of 89.76%, but the ALTS-MLODL, DL-MNN, ANN, KELM, and ELM systems provided lower $sens_y$ values of 86.54%, 83.60%, 81.79%, 80.72%, and 78.98%, respectively. Moreover, on a 5000kb file, the ETS-NLPODL system gave a greater $sens_y$ of 99.23%. However, the ALTS-MLODL, DL-MNN, ANN, KELM, and ELM methodologies offered lesser $sens_y$ values of 98.98%, 95.79%, 94.71%, 93.69%, and 92.41%, correspondingly.

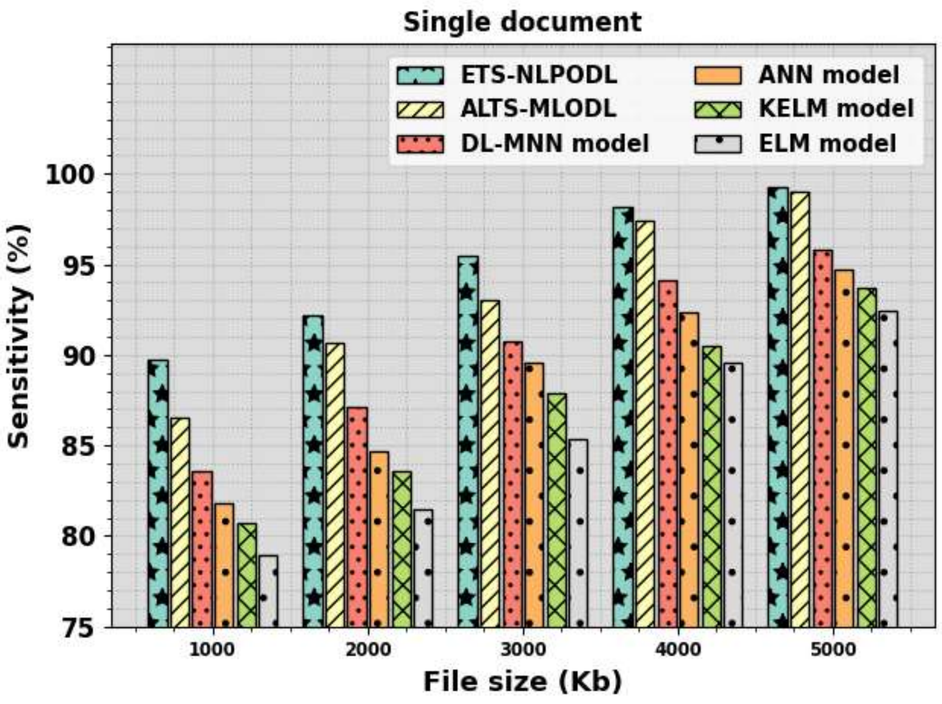

Figure 3. $Sens_y$ analysis of the ETS-NLPODL algorithm under a single document.

Table 1. Comparison analysis of the ETS-NLPODL technique with other models on a single document.

Single document						
File size (Kb)	**ETS-NLPODL**	**ALTS-MLODL**	**DL-MNN model**	**ANN model**	**KELM model**	**ELM model**
Sensitivity						
1000	89.76	86.54	83.60	81.79	80.72	78.98
2000	92.16	90.70	87.12	84.64	83.61	81.47
3000	95.49	93.06	90.71	89.53	87.88	85.35
4000	98.15	97.42	94.16	92.36	90.46	89.54
5000	99.23	98.98	95.79	94.71	93.69	92.41
Specificity						
1000	89.32	86.00	83.21	80.75	79.62	78.16
2000	90.01	88.89	85.55	83.25	81.91	79.53
3000	93.97	92.08	88.96	86.60	85.58	83.94
4000	96.13	94.46	91.88	90.81	89.29	88.10
5000	97.25	95.58	93.27	91.08	88.51	86.86
Accuracy						
1000	89.36	87.75	84.57	82.51	80.08	78.46
2000	92.10	90.68	87.77	85.27	84.11	82.05
3000	94.19	92.94	90.26	88.68	87.07	85.30
4000	97.80	96.00	92.25	90.01	88.38	86.74
5000	98.85	98.27	94.74	92.60	90.22	88.90
F-score						
1000	89.96	88.01	84.40	82.87	81.62	79.99
2000	91.00	89.91	87.24	85.27	83.74	82.65
3000	93.58	92.06	89.35	88.21	86.76	85.44
4000	96.49	95.55	92.89	91.91	89.54	87.83
5000	98.87	97.31	94.88	93.27	91.71	90.25

Figure 4 illustrates a $spec_y$ analysis of the ETS-NLPODL system with other systems. The obtained outcomes indicate that the ETS-NLPODL method attained enriched outcomes with different file sizes. According to the 1000kb file, the ETS-NLPODL methodology gave a greater $spec_y$ of 89.32% but the ALTS-MLODL, DL-MNN, ANN, KELM, and ELM algorithms offered reduced $spec_y$values of 86.00%, 83.21%, 80.75%, 79.62%, and 78.16%, respectively. Also, with a 5000kb file, the ETS-NLPODL system gave a greater $spec_y$ of 97.25%. However, the ALTS-MLODL, DL-MNN, ANN, KELM, and ELM methodologies obtained lesser $sens_y$ values of 95.58%, 93.27%, 91.08%, 88.51%, and 86.86%, respectively.

Figure 5 displays an $accu_y$ analysis of the ETS-NLPODL system with other techniques. The achieved outcomes show that the ETS-NLPODL algorithm received improved outcomes with all file sizes. Additionally, with a 1000kb file, the ETS-NLPODL method gave boosted $accu_y$ of 89.36%, and the ALTS-MLODL, DL-MNN, ANN, KELM, and ELM systems offered reduced $accu_y$values of 87.75%, 84.57%, 82.51%, 80.8%, and 78.46%. Besides, with a 5000kb file, the ETS-NLPODL system gives a better $accu_y$ of 98.85%, while the ALTS-MLODL, DL-MNN, ANN, KELM, and ELM systems acquired decreased $accu_y$ values of 98.27%, 94.74%, 92.60%, 90.22%, and 88.90%, respectively.

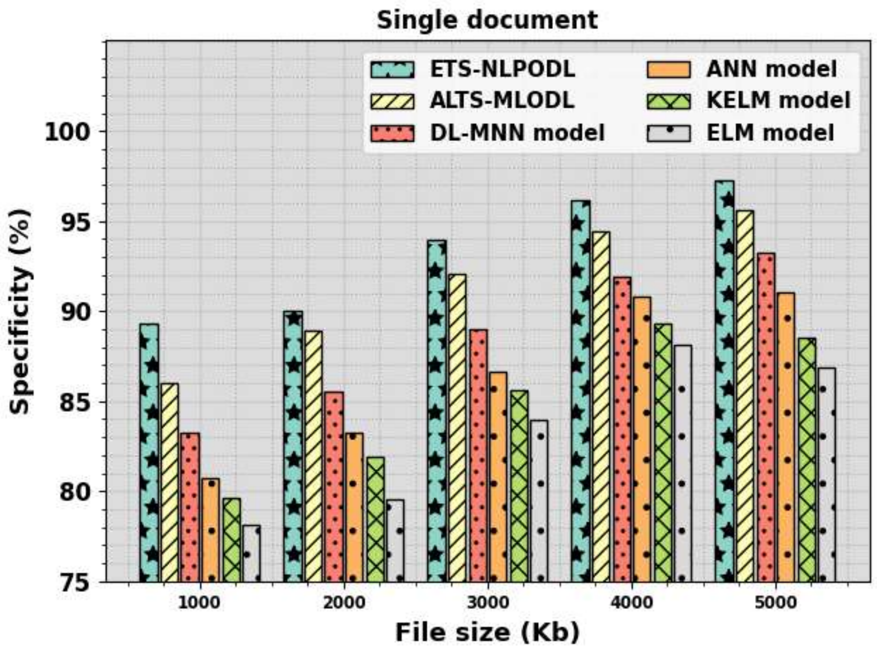

Figure 4. $Spec_y$ analysis of the ETS-NLPODL algorithm under a single document.

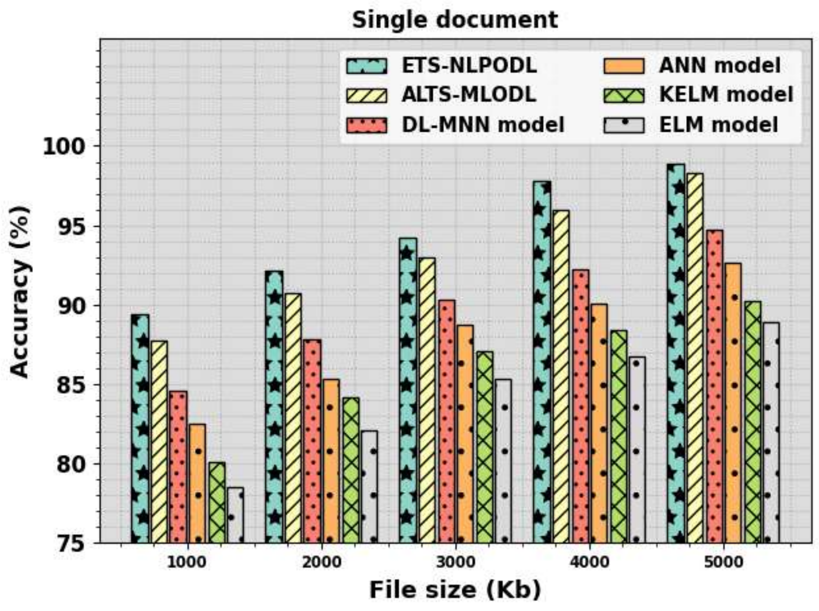

Figure 5. $Accu_y$ analysis of the ETS-NLPODL model with a single document.

Figure 6 displays an F_{score} analysis of the ETS-NLPODL algorithm with existing models. The achieved outcomes show that the ETS-NLPODL method received increased outcomes with file sizes.

According to the 1000kb file, the ETS-NLPODL approach gave boosted F_{score} of 89.96%, whereas the ALTS-MLODL, DL-MNN, ANN, KELM, and ELM algorithms had reduced F_{score} values of 88.01%, 84.40%, 82.87%, 81.62%, and 79.99%. Further, on a 5000kb file, the ETS-NLPODL system gave a higher F_{score} of 98.87%, and the ALTS-MLODL, DL-MNN, ANN, KELM, and ELM methodologies acquired decreased F_{score} values of 97.31%, 94.88%, 93.27%, 91.71%, and 90.25%, correspondingly.

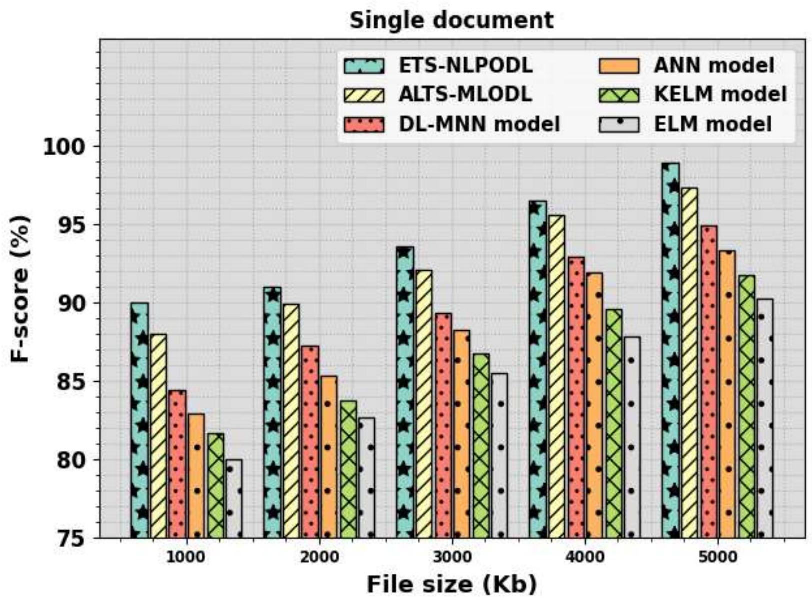

Figure 6. F_{score} analysis of the ETS-NLPODL algorithm under a single document.

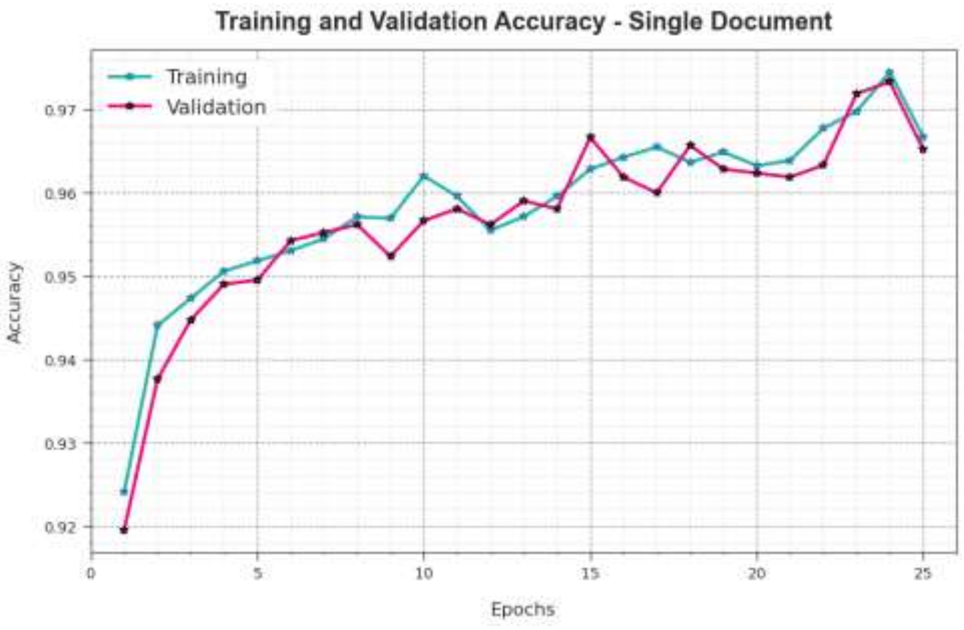

Figure 7. $Accu_y$ curve of the ETS-NLPODL model under a single document.

The $accu_y$ of training (TRA) and validation (VL) curves of the ETS-NLPODL system under a single document exhibited in Figure 7 offers valued insights into the effectiveness of the ETS-NLPODL method over numerous epochs. These curves show crucial insights into the learning development and capability of the model in generalization. Moreover, it could be noticed, that it has a consistency enhancement in the TRA and TES $accu_y$ over increasing epochs. This exhibits the capacity of the model for learning and recognizing patterns within the datasets of TRA and TES. The improving TES $accu_y$ shows that the model not only adjusts to the TRA data but excels to make accurate predictions on earlier unseen data, highlighting the capabilities of robust generalization.

In Figure 8, we denote an extensive outcome of the TR (Training) and TS (Testing) loss values for the ETS-NLPODL model with a single document through diverse epochs. The TRA loss gradually decreases as the model enhances its weights for decreasing classification errors under both datasets of TRA and TES. These loss curves offer a perfect image of how well the model aligns with the TRA data, emphasizing its capability for proficiently holding the patterns in these datasets. It is valuable to observe that the ETS-NLPODL system continually improves its parameters to minimize the discrepancies between the prediction and actual TRA labels.

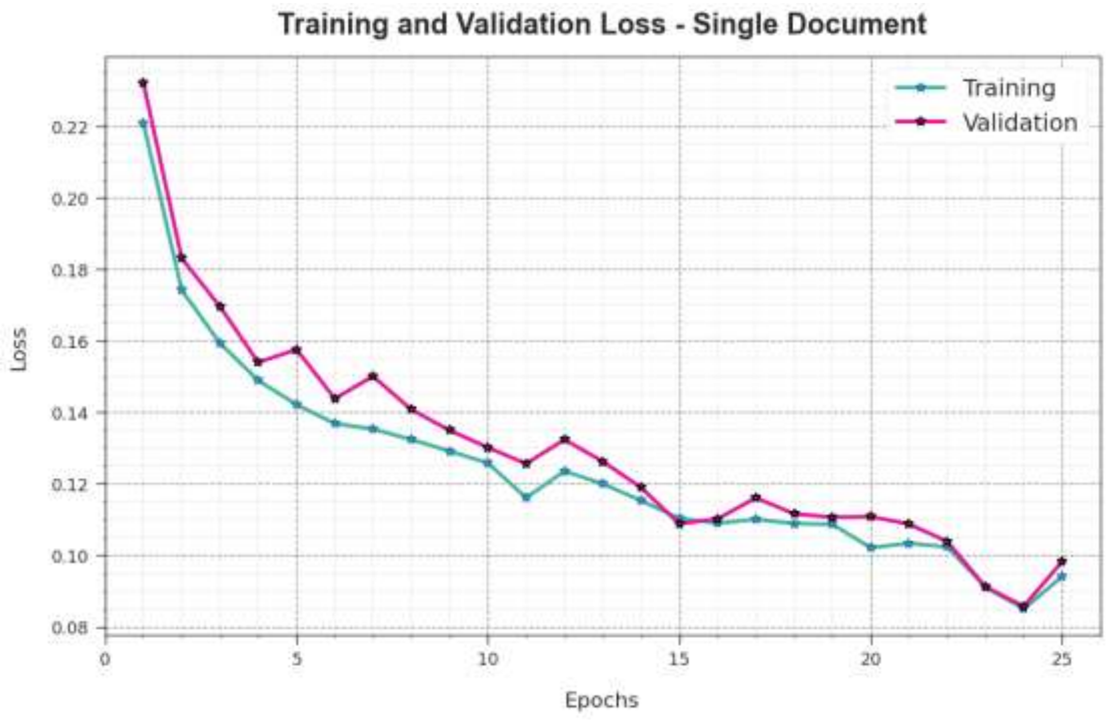

Figure 8. Loss curve of the ETS-NLPODL approach with a single document.

Table 2 reports a comprehensive result analysis of the ETS-NLPODL technique on multi-documents. In Figure 9, the $sens_y$ results of the ETS-NLPODL technique are investigated under varying file sizes. The accomplished outcomes display that the ELM and KELM models have reported poor performance, whereas the ANN model reaches slightly increased results. Along with that, the DL-MNN and ALTS-MLODL models offer considerable performance. But the ETS-NLPODL technique gains maximum performance with a $sens_y$ of 89.01%, 91.96%, 94.20%, 96.58%, and 98.37% under file sizes of 1000–5000Kb, respectively.

In Figure 10, the $spec_y$ outcomes of the ETS-NLPODL system can be examined under changing file sizes. The achieved outcomes display that the ELM and KELM methods had inferior performance while the ANN technique acquired moderately boosted outcomes. In addition, the DL-MNN and ALTS-MLODL methodologies gave significant performance. However, the ETS-NLPODL algorithm attained great performance with $spec_y$ of 88.13%, 89.78%, 91.09%, 92.48%, and 96.99% on file sizes of 1000–5000Kb, correspondingly.

In Figure 11, the $accu_y$ outcomes of the ETS-NLPODL system can be examined under changing file sizes. The achieved outcomes display that the ELM and KELM methods had lower performance while the ANN technique acquired moderately boosted outcomes. In addition, the DL-MNN and ALTS-MLODL methodologies gave significant performance. However, the ETS-NLPODL algorithm attained great performance with $accu_y$ of 89.57%, 93.00%, 96.00%, 97.89%, and 98.97% for file sizes of 1000–5000Kb.

Table 2. Comparative outcomes of the ETS-NLPODL technique with other approaches under multi-documents.

Multi-document						
File size (Kb)	**ETS-NLPODL**	**ALTS-MLODL**	**DL-MNN model**	**ANN model**	**KELM model**	**ELM model**
Sensitivity						
1000	89.01	87.06	84.05	82.05	80.30	78.51
2000	91.96	89.38	86.49	84.93	83.20	80.83
3000	94.20	92.33	89.23	86.83	85.08	83.10
4000	96.58	95.18	91.99	90.00	88.61	87.49
5000	98.37	97.67	94.21	92.70	90.44	89.41
Specificity						
1000	88.13	86.71	83.78	82.55	80.61	78.55
2000	89.78	87.62	85.07	82.82	80.80	79.58
3000	91.09	89.24	86.66	85.18	82.58	81.48
4000	92.48	90.22	87.56	85.40	84.40	82.40
5000	96.99	95.96	90.13	88.12	86.21	84.00
Accuracy						
1000	89.57	87.15	84.46	83.20	81.93	80.39
2000	93.00	91.25	88.21	86.53	84.62	83.53
3000	96.00	94.85	91.49	89.06	87.61	85.52
4000	97.89	96.57	92.93	90.85	89.15	87.28
5000	98.97	98.21	96.45	94.18	92.14	90.77
F-score						
1000	89.78	88.05	84.70	82.14	80.25	77.76
2000	91.69	90.67	87.65	86.31	84.70	83.16
3000	94.28	93.59	90.68	89.13	87.08	84.89
4000	98.12	97.05	94.09	91.95	89.95	88.63
5000	99.07	98.76	97.61	95.61	94.65	93.33

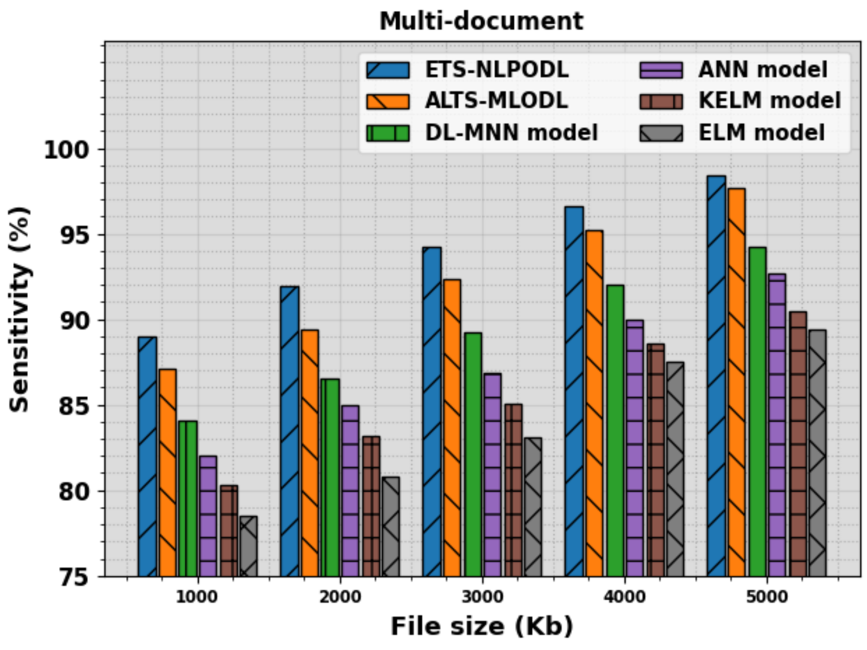

Figure 9. $Sens_y$ analysis of the ETS-NLPODL model under multi-documents.

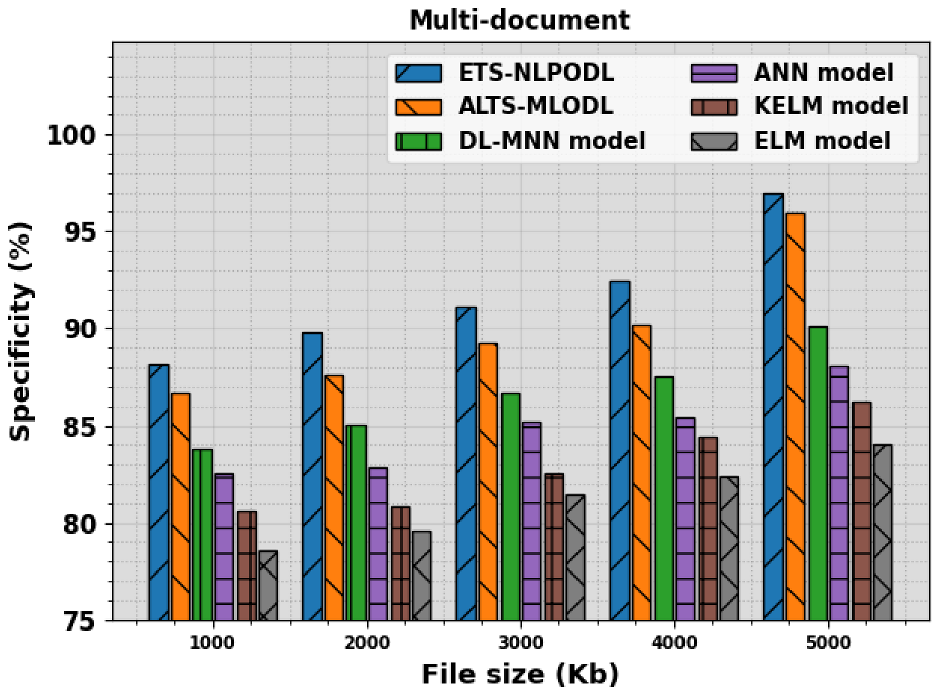

Figure 10. $Spec_y$ analysis of the ETS-NLPODL system with multi-documents.

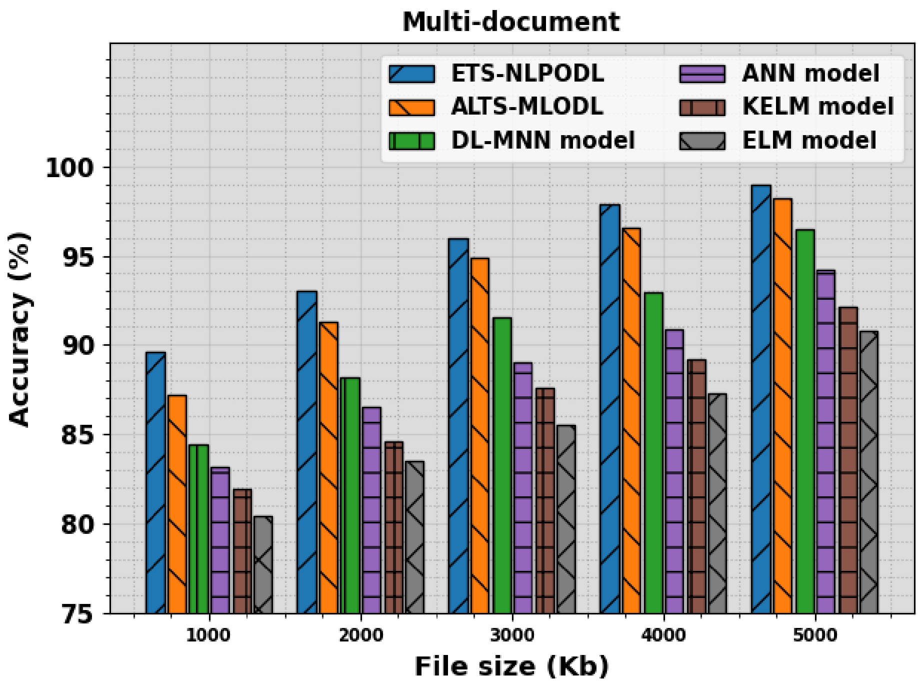

Figure 11. $Accu_y$ analysis of the ETS-NLPODL algorithm under multi-documents.

In Figure 12, the F_{score} outcomes of the ETS-NLPODL method can be confirmed with various file sizes. The accomplished outcomes show the ELM and KELM algorithms were defined as having lower performance, whereas the ANN technique got reasonably boosted outcomes. The DL-MNN and ALTS-MLODL systems gave significant performance. However, the ETS-NLPODL methodologies achieved better performance with F_{score} of 89.78%, 91.69%, 94.28%, 98.12%, and 99.07% with file sizes of 1000–5000Kb.

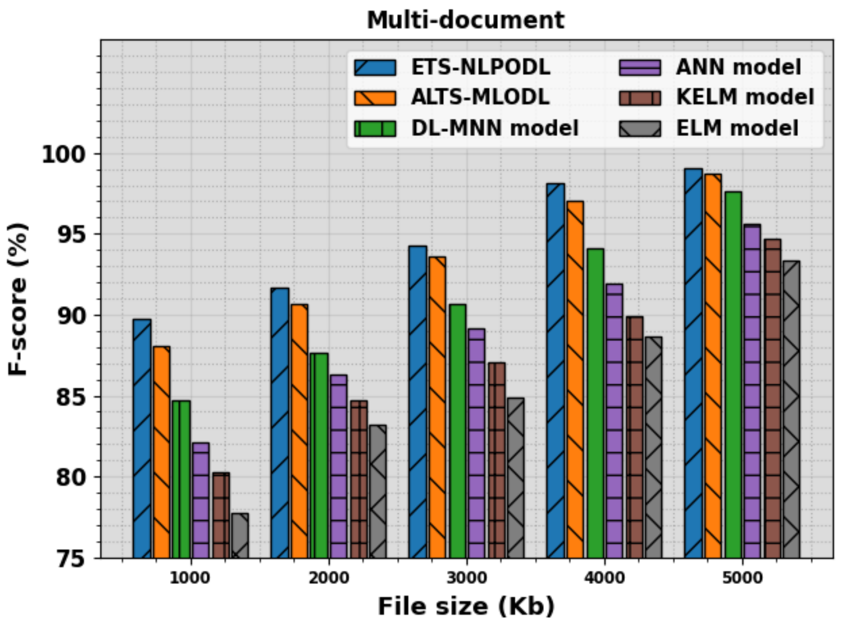

Figure 12. F_{score} outcome of the ETS-NLPODL model under multi-documents.

The training (TRA) and validation (VL) $accu_y$ curves of the ETS-NLPODL system under multi-documents displayed in Figure 13 offer valued insights into the effectiveness of the ETS-NLPODL algorithm over multiple epochs. These curves display vital insights into the learning expansion and capability of the model for generalization. Additionally, it must be observed, that which has a constancy enrichment in the TRA and TES $accu_y$ over rising epochs. It reveals the capacity of the model for recognizing and learning patterns with these datasets of TRA and TES. The enhancement TES $accu_y$ shows that the model not only changes to the TRA data but also surpasses to produce correct predictions on prior unnoticed data, emphasizing the capabilities of robust generalization.

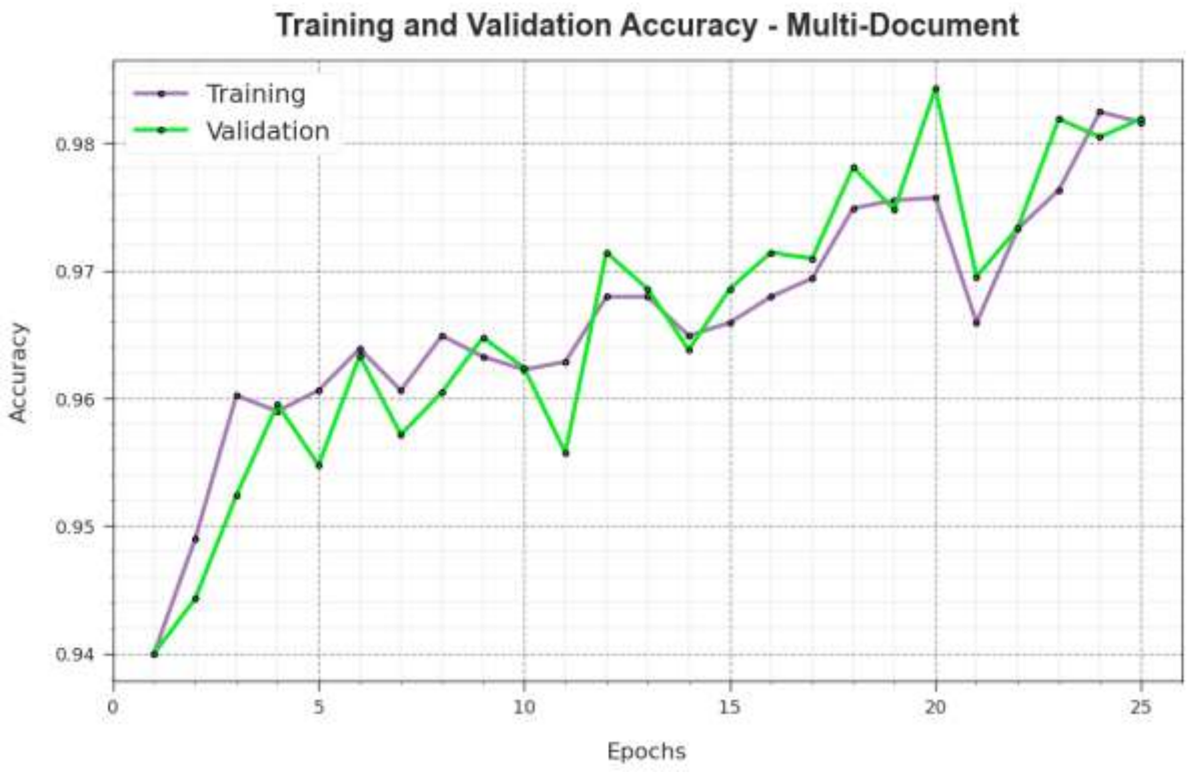

Figure 13. $Accu_y$ curve of the ETS-NLPODL method under multi-documents.

In Figure 14, we indicate an extensive view of the Training (TRA) and Testing (TES) loss values for the ETS-NLPODL technique with multi-documents in various epochs. The TRA loss slowly decreased as the model boosted its weights to minimize classification errors under both datasets of TRA and TES. These loss curves provide a precise perception of how the model can align with the training data, emphasizing its capability for proficiently holding the patterns in these datasets. It is valuable to observe that the ETS-NLPODL methodology continually increased its parameters to lessen the differences between the predicted and actual TRA labels.

The achieved outcomes ensure that the ETS-NLPODL system had better performance than other models.

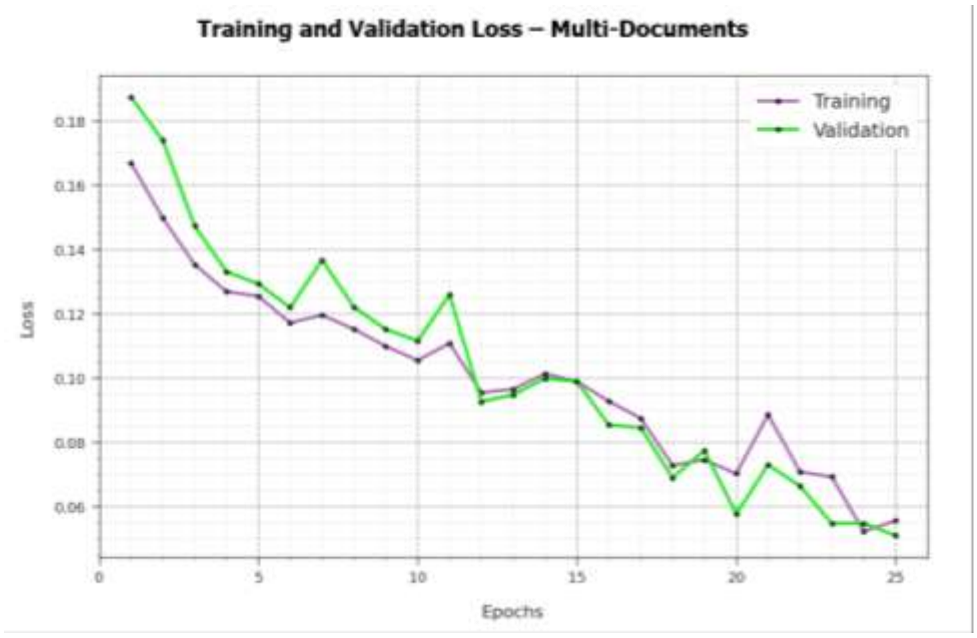

Figure 14. Loss curve of the ETS-NLPODL model on multi-documents.

5. Conclusions

In this manuscript, we focused on the designs and development of the ETS-NLPODL methodology. The major goal of the ETS-NLPODL technique was to exploit feature selection with a hyperparameter-tuned DL model for summarizing the text. In the ETS-NLPODL technique, an initial step of data preprocessing was involved to convert the input text into a compatible format. Next, the feature extraction process was carried out and the optimal set of features was chosen by the HGSO algorithm. For text summarization, the ETS-NLPODL model used the ACNN-GRU model. Finally, the MGO algorithm was implemented for the optimal hyperparameter selection of the ACNN-GRU system. The achieved outcomes of the ETS-NLPODL technique were tested on a benchmark dataset. The experimentation outcomes pointed out that the ETS-NLPODL method gained better performance over other models concerning diverse performance measures.

Use of AI tools declaration

The authors declare that they have not used artificial intelligence (AI) tools in the creation of this article.

Acknowledgments

The authors extend their appreciation to the Deanship of Scientific Research at King Khalid University for funding this work through a large group research project under grant number (RGP2/185/44); Princess Nourah bint Abdulrahman University Researchers Supporting Project number (PNURSP2024R263), Princess Nourah bint Abdulrahman University, Riyadh, Saudi Arabia; and Research Supporting Project number (RSPD2024R787), King Saud University, Riyadh, Saudi

Arabia. The authors extend their appreciation to the Deanship of Scientific Research at Northern Border University, Arar, KSA, for funding this research work through the project number NBU-FPEJ-2024- 2903-01. The research team would like to acknowledge Prince Sultan University for paying the journal's APC. This study is partially funded by the Future University in Egypt (FUE).

Conflict of interest

The authors declare that they have no conflicts of interest. The manuscript was written through the contributions of all authors. All authors have approved the final version of the manuscript.

References

1. M. Yadav, R. Katarya, A Systematic Survey of Automatic Text Summarization Using Deep Learning Techniques, In *Modern Electronics Devices and Communication Systems: Select Proceedings of MEDCOM 2021*, 397–405. Singapore: Springer Nature Singapore, 2023. https://doi.org/10.1007/978-981-19-6383-4_31
2. Y. M. Wazery, M. E. Saleh, A. Alharbi, A. A. Ali, Abstractive Arabic text summarization based on deep learning, *Comput. Intel. Neurosci.*, 2022. https://doi.org/10.1155/2022/1566890
3. P. J. Goutom, N. Baruah, P. Sonowal, An abstractive text summarization using deep learning in Assamese, *Int. J. Inf. Technol.*, 2023, 1–8. https://doi.org/10.1007/s41870-023-01279-7
4. V. L. Sireesha, *Text Summarization for Resource-Poor Languages: Datasets and Models for Multiple Indian Languages* (Doctoral dissertation, International Institute of Information Technology Hyderabad), 2023.
5. S. Dhankhar, M. K. Gupta, Automatic Extractive Summarization for English Text: A Brief Survey. In *Proceedings of Second Doctoral Symposium on Computational Intelligence: DoSCI 2021*, 183–198. Singapore: Springer Singapore, 2021. https://doi.org/10.1007/978-981-16-3346-1_15
6. B. Shukla, S. Gupta, A. K. Yadav, D. Yadav, Text summarization of legal documents using reinforcement learning: A study, In *Intelligent Sustainable Systems: Proceedings of ICISS 2022*, 403–414. Singapore: Springer Nature Singapore, 2022. https://doi.org/10.1007/978-981-19-2894-9_30
7. B. Baykara, T. Güngör, Turkish abstractive text summarization using pretrained sequence-to-sequence models, *Nat. Lang. Eng.*, **29** (2023), 1275–1304. https://doi.org/10.1017/S1351324922000195
8. M. Bani-Almarjeh, M. B. Kurdy, Arabic abstractive text summarization using RNN-based and transformer-based architectures, *Inform. Process. Manag.*, **60** (2023), 103227. https://doi.org/10.1016/j.ipm.2022.103227
9. H. Aliakbarpour, M. T. Manzuri, A. M. Rahmani, Improving the readability and saliency of abstractive text summarization using a combination of deep neural networks equipped with auxiliary attention mechanisms, *J. Supercomput.*, 2022, 1–28.
10. S. N. Turky, A. S. A. Al-Jumaili, R. K. Hasoun, Deep learning based on different methods for text summary: A survey, *J. Al-Qadisiyah Comput. Sci. Math.*, **13** (2021), 26. https://doi.org/10.29304/jqcm.2021.13.1.766
11. G. A. Babu, S. Badugu, Deep learning based sequence to sequence model for abstractive Telugu text summarization, *Multimed. Tools Appl.*, **82** (2023), 17075–17096. https://doi.org/10.1007/s11042-022-14099-x

12. S. A. Tripathy, A. Sharmila, Abstractive method-based text summarization using bidirectional long short-term memory and pointer generator mode, *J. Appl. Res. Technol.*, **21** (2023), 73–86. https://doi.org/10.22201/icat.24486736e.2023.21.1.1446
13. N. Shafiq, I. Hamid, M. Asif, Q. Nawaz, H. Aljuaid, H. Ali, Abstractive text summarization of low-resourced languages using deep learning, *PeerJ Comput. Sci.*, **9** *(*2023), e1176. https://doi.org/10.7717/peerj-cs.1176
14. R. Karmakar, K. Nirantar, P. Kurunkar, P. Hiremath, D. Chaudhari, Indian regional language abstractive text summarization using attention-based LSTM neural network, In *2021 International Conference on Intelligent Technologies (CONIT)*, 1–8, IEEE, 2021. https://doi.org/10.1109/CONIT51480.2021.9498309
15. R. Rani, D. K. Lobiyal, Document vector embedding based extractive text summarization system for Hindi and English text, *Appl. Intell.*, 2022, 1–20. https://doi.org/10.1007/s10489-021-02871-9
16. W. Etaiwi, A. Awajan, SemG-TS: Abstractive Arabic text summarization using semantic graph embedding, *Mathematics*, **10** (2022), 3225. https://doi.org/10.3390/math10183225
17. R. T. AlTimimi, F. H. AlRubbiay, Multilingual text summarization using deep learning, *Int. J. Eng. Adv. Technol.*, **7** (2021), 29–39. https://doi.org/10.31695/IJERAT.2021.3712
18. S. V. Moravvej, A. Mirzaei, M. Safayani, Biomedical text summarization using conditional generative adversarial network (CGAN), *arXiv preprint arXiv:2110.11870*, 2021.
19. A. Al Abdulwahid, Software solution for text summarisation using machine learning based Bidirectional Encoder Representations from Transformers algorithm, *IET SOFTWARE*, 2023. https://doi.org/10.1049/sfw2.12098
20. B. Muthu, S. Cb, P. M. Kumar, S. N. Kadry, C. H. Hsu, O. Sanjuan, et al., A framework for extractive text summarization based on deep learning modified neural network classifier, ACM *T. Asian Low-Reso.*, **20** (2021), 1–20. https://doi.org/10.1145/3392048
21. D. Izci, S. Ekinci, E. Eker, M. Kayri, Augmented hunger games search algorithm using logarithmic spiral opposition-based learning for function optimization and controller design, *J. King Saud University-Eng. Sci.*, 2022. https://doi.org/10.1016/j.jksues.2022.03.001
22. M. Mafarja, T. Thaher, M. A. Al-Betar, J. Too, M. A. Awadallah, I. Abu Doush, et al., Classification framework for faulty-software using enhanced exploratory whale optimizer-based feature selection scheme and random forest ensemble learning, *Appl. Intell.*, 2023, 1–43. https://doi.org/10.1007/s10489-022-04427-x
23. B. Liu, J. Xu, W. Xia, State-of-Health Estimation for Lithium-Ion Battery Based on an Attention-Based CNN-GRU Model with Reconstructed Feature Series, *Int. J. Energy Res.*, 2023. https://doi.org/10.1155/2023/8569161
24. P. Sarangi, P. Mohapatra, Evolved opposition-based Mountain Gazelle Optimizer to solve optimization problems, *J. King Saud Univ-Com.*, 2023, 101812. https://doi.org/10.1016/j.jksuci.2023.101812
25. H. J. Alshahrani, K. Tarmissi, A. Yafoz, A. Mohamed, M. A. Hamza, I. Yaseen, et al., Applied linguistics with mixed leader optimizer based English text summarization model, *Intell. Autom. Soft Co.*, **36** (2023). https://doi.org/10.32604/iasc.2023.034848