ISSN 0147-6882, Scientific and Technical Information Processing, 2024, Vol. 51, No. 1, pp. 72–83. © Allerton Press, Inc., 2024.

A Hybrid Approach for Automatic Text Summarization by Handling Out-of-Vocabulary Words Using TextR-BLG Pointer Algorithm

Sonali Mhatre[a], * and Lata L. Ragha[b]

[a] *Department of Computer Engineering, Terna Engineering College, Nerul, Navi Mumbai, Maharashtra, 400706 India*

[b] *Department of Computer Engineering, Fr. Conceicao Rodrigues Institute of Technology, Vashi, Navi Mumbai, Maharashtra, 400703 India*

**e-mail: sonalinmhatre@gmail.com*

Received November 23, 2023

Abstract—Long documents such as scientific papers and government reports, often discuss substantial issues in long conversation, which are time-consuming to read and understand. Generating abstractive summaries can help readers quickly grasp the main topics, yet prior work has mostly focused on short texts and also has some of the drawbacks such as out-of-vocabulary (OOV), mismatched sentences and meaning less summary. Hence, to overcome these issues the hybrid approach for automatic text summarization using the TextR-BLG pointer algorithm. In this designed model, the long document is given as an input for automatic text summarization and are evaluated for the word frequency length, based on the threshold value the sentences are split into extractive and abstractive approach. The text algorithm used in this model finds out the sentence similarity score and is validated with the plotted graph. Likewise, the sentences above the threshold level are considered as the abstractive approach. The optimized BERT-LSTM-BiGRU (BLG) based pointer algorithm is used for learning the meaning from the sentences by word embedding, encoding and decoding the hidden states. Finally, the reframed sentences are considered for attaining the similarity score and plotting graph. The sentences from the abstractive and extractive approach are ranked based on the plotted graph for generating the summary. For evaluating the performance of the model, the ROUGE 1, ROUGE 2, ROUGE L, Bert Score, Bleu Score, and Meteor Score are 59.2, 58.4, 62.3, 0.92, 0.78, and 0.67, which are compared with the existing model. From the evaluation of proposed and existing summarization techniques, the proposed model attains better text summarization than the existing model. Thus, the hybrid approach for automatic text summarization using the TextR-BLG pointer algorithm performs better by handling out-of-vocabulary words than the existing text summarization techniques.

Keywords: text summarization, text rank, pointer generator algorithm, ROUGE score, OOV words, BERT

DOI: 10.3103/S0147688224010106

INTRODUCTION

Text summarization is one type of natural language processing (NLP) operation that are crucial because they enable humans to quickly learn crucial information from a huge number of texts [1]. Large texts are particularly challenging for humans to manually summarise but there is an abundance of text material available on the internet. However, the Internet provides usually more information than needed [2]. As a result, there are two issues that must be solved such as identifying significant materials among the vast amount of documents that are available and taking in an extensive amount of useful information. A summary may be used to lead readers to specific passages in the original document or it may be used to cover all the pertinent details in the text [3]. The shorter reading time of a summary has become the main advantage in both situations. A decent summary system should reflect the document's variety of themes while minimizing redundancy. In order to find the main ideas of a document, summarization algorithms may additionally look for headers and other indicators of subtopics [4]. The AutoSummarize feature in Microsoft Word is a straightforward illustration of text summarising.

Abstractive summarization and Extractive summarization are the two categories of text summarising techniques. An extractive summary approach involves taking important sentences, paragraphs, etc., from the source content and merging them into a shorter form [5]. Sentences are ranked according to their linguistic and statistical properties. An abstractive summary aims to comprehend the key ideas in a document and to communicate those ideas in an understandable common language [6]. It employs linguistic techniques to analyze and understand the text before identifying new terms and idioms that effectively capture its meaning and creating a new, shorter text that highlights the most crucial details from the original text document

[7]. After encoding a complete document, neural-based abstractive models often use a seq2seq framework to provide a word-by-word summary. Extractive models, in contrast, source text is directly choose significant passages and then combined into a summary [8]. While extractive models are more efficient and factually accurate, abstractive models are typically more flexible and may result in difficult or incorrect summary sentences.

Despite their accomplishment, simulating long-range inter-sentence interactions for summarization is still difficult [9]. By treating a document as a series of sequences, hierarchical networks are typically used to address this issue. However, empirical findings indicated that using a framework like this to describe intersentence interactions did not significantly improve performance for summarising. In addition to being slow to train, hierarchical techniques frequently overfit [10]. Most recently, the modelling of cross-sentence interactions for summarization tasks has received extensive attention through the use of graph neural networks (GNNs). Establishing a strong document graph is essential for this framework to work. Discourse analysis was used in several research to create document graphs. However, this method relies on outside resources and could provide output that is semantically collapsed among other issues. Hence, in this model, for summarizing the long documents, both the extractive and abstractive approach such as the text rank algorithm and pointer generator algorithm based on BERT, LSTM and GRU, which are combined together to form a hybrid (TextR-BLG) generator algorithm. Major contributions of the proposed hybrid TextR-BLG generator algorithm are listed below.

• For summarizing long documents, hybrid extractive and abstractive approach are combined together.

• Word length frequencies are calculated and based on the threshold, the document sentences are considered as extractive approach or abstractive approach.

• Text rank algorithm is used for ranking the sentences in the paragraph using a graph-based ranking algorithm for text summarization.

• BERT algorithm used in this model is a word embedding layer for vector representations of words that are learned during the pre-training of the BERT model.

• Out-of-vocabulary (OOV) words are handled by using the abstractive pointer generator algorithm using the BERT, LSTM and Bi-GRU.

• LSTM and Bi-GRU parameters are optimally selected using the squid game optimizer for enhancing the learning rate of the algorithm.

The upcoming portions of the paper are illustrated as section 2 consists of various articles related to the text summarization and section 3 briefly explains the proposed hybrid TextR-BLG generator algorithm. Section 4 describes about the experimental outcomes and section 5 concludes the entire article.

LITERATURE SURVEY

Text summarization is mainly used to solve storage issues due to the massive development of data and also to reduce the time of the readers. Many articles are introduced and implemented for automatic text summarization using extractive and abstractive approach. Some of the articles related to text summarization are studied and reviewed below.

A general framework for guided neural abstractive summarization was developed by Dou et al. [11]. Although neural abstractive summarising models are adaptable and capable of creating coherent summaries, they can also be unreliable at times and are challenging to manage. A generic and expandable guided summarization framework (GSum) in this existing architecture may successfully accept various forms of external guidance as input and carry out experiments across a variety of distinct variations. The model achieves state-of-the-art performance on four well-known summarization datasets in experiments when utilizing highlighted sentences as training examples, according to ROUGE.

Fine-tune BERT for extractive summarization was developed by Liu [12]. A pretrained Transformer model named BERT has excelled in a number of NLP tasks. The extractive summarization task was the main emphasis of the several BERT model versions employed in this model, and the CNN/Dailymail and NYT datasets are used to analyze. Each sentence must have an output representation to use BERT for extractive summarization. As a masked-language model, BERT was trained to produce output vectors that are based on tokens instead of sentences. BERT employs just two labels, while extractive summarization employs plenty of labels. This was the various sentences are indicated by segmentation embedding in extractive summarization. As a result, the input sequence and BERT embedding of the existing model were updated to allow for summary extraction.

Discourse-aware neural extractive text summarization was developed by Xu et al. [13]. Recent advances in text summarization models have utilized BERT for document encoding. However, the extracted summaries from sentence-based extractive algorithms frequently contain sentences that are unnecessary or uninformative. The BERT algorithm was not pre-trained on documents, but rather on phrase pairings does a poor job of capturing long-range dependencies across a document. A discourse-aware neural summarization model called DISCOBERT1 was used to address these problems since it separates sub-sentential discourse units (instead of sentences) as possible options for extractive selection on a finer granularity. In order to encode co-reference mentions and capture

long-term dependencies between discourse components, hierarchical discourse graphs are built using RST trees and graph convolutional networks.

Text summarization based on clustering and optimization was developed by Alguliyev et al. [14]. This was a promising field because current research on text summarization according to optimization, evolutionary algorithms, and clustering algorithms has produced positive results. This existing approach develops a two-stage phrase selection model for text summarising termed as COSUM, which was based on clustering and optimization techniques. The first step involves utilizing the *k*-means approach to cluster the set of phrases in order to identify all of the text's subjects. An optimization model was used in the second stage to identify the most important sentences from clusters. In this paradigm, an objective function, represented as the harmonic mean of the objective functions, enforces the diversity and coverage of the chosen sentences in the summary. An adaptive differential evolution algorithm with a cutting-edge mutation method was created to address the optimization challenge.

Abstractive email thread summarization was developed by Zhang et al. [15]. An abstractive email thread summarization (EMAILSUM) mode was created using the Email dataset, which have 2.549 email chats were compiled into long (100 words) and short (thirty words) summaries with human annotation over a wide range of topics to encourage research in thread summarization. A comprehensive empirical study has been developed to investigate various summarization strategies such as single-document, abstractive and extractive methods and hierarchical methods to carry out human evaluations on long and short summary generation process. Identifying the sender's purpose and the sender and receiver's roles, which are the main problems with the current abstractive summarization models, are solved by this method. Additionally, for this task of summarising an email thread, the frequently used automatic assessment metrics (ROUGE, BERTScore) have a limited correlation with human judgements.

Above reviewed articles are related to the extractive and abstractive approach for text summarization. Most of the above mentioned techniques attain better performance for text summarization, yet contains some of the drawbacks that needs to be addressed. The GSUM model used for text summarization consists of the BART model, which is based on the abstractive approach lags for the long document to summarize [11]. Fine-tune BERT for extractive summarization can be easy to summarize but the out-of-vocabulary (OOV) words are not managed and also it can be repetitive, incoherent or miss important information [12, 13]. Discourse-aware neural extractive text summarization based on MLP has hidden layers, which have a non-convex loss function that exists more than one local minimum. Therefore, different random weight initializations can lead to different validation accuracy and also it is sensitive to feature scaling [14]. *k*-means clustering algorithm [15] is sensitive to initial conditions, difficulty in determining, inability to handle categorical data.

PROPOSED METHODOLOGY

Creating a summary of content that is popular helps consumer to easily search for information and obtain preliminary facts in a short period of time. Despite the fact that numerous scientific writings are being generated and published in journals and conferences, data growth resulted in the development of new sciences in a variety of sectors. The conventional summary method fails to be an effective solution to this issue. Hence, in this model, both the abstractive and extractive approach are combined together for accurately summarizing the long documents by eliminating the OOV issues. Figure 1 illustrates the process flow of the hybrid text summarization algorithm.

In this designed model, the long document is given as an input for automatic text summarization based on the hybrid extractive and abstractive approach. The sentence in the document are evaluated for the word frequency length and based on the threshold values the sentences are split into two processing approaches such as extractive and abstractive approach. The text algorithm used in this model finds out the sentence similarity score and is validated with the plotted graph. Likewise, the sentences above the threshold level are considered as the abstractive approach. The BERT-LSTM-BiGRU (BLG) based generator algorithm is used for learning the meaning from the sentences by word embedding, encoding and decoding the hidden states. The learning rate, batch size, number of neurons and activation function are optimally selected using the squid game optimizer. Finally, the reframed sentence from the BLG generator algorithm is considered for attaining the similarity score and plotting graph. The sentences are ranked based on the plotted graph for generating the summary. Based on this approach, the OOV problems are solved by embedding the OOV word sentences with similar sentences based on the pretrained model.

Optimized TextR-BLG Generator Algorithm

In this proposed TextR-BLG generator algorithm, the extractive approach, such as the text rank algorithm and the abstractive approach, such as the BERT-LSTM-BiGRU (BLG) based generator algorithm. These extractive and abstractive approaches are combined together for generating the summary.

Text Rank Algorithm

Text Rank algorithm uses the PageRank algorithm to compare sentences in a document to web pages. The

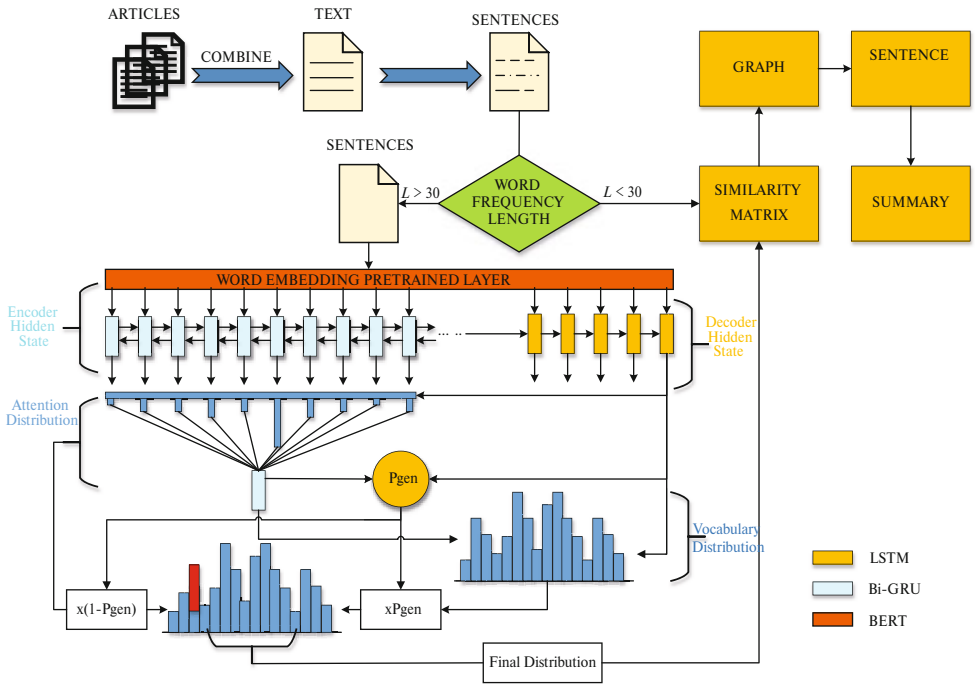

Fig. 1. Architecture of TextR-BLG Generator Algorithm.

degree of resemblance between the two sentences determines the probability that sentence A will lead to sentence B. The Text Rank algorithm [16] ranks sentences using the same logic as the PageRank algorithm, allowing users to choose the most significant sentences from the input text content. Pages with high PageRank are connected to other high PageRank pages. The essential sentences in this method are connected to additional significant sentences in the input document in a similar manner. The sentences in this model are identified by converting the input document M into the sentences $\{S_1, S_2, S_3 \ldots, S_n\}$, i.e., $M = \{S_1, S_2, S_3 \ldots, S_n\}$. Considering the sentence in the document as an input sequence, an index is allocated to each sentence. In order to properly arrange the sentences that make up the summary and give it meaning, index values are frequently utilized. Each sentence is then tokenized into a group of terms in the following phase. In the given text document, every sentence has been expressed by a word vector in order to define sentence similarity. The next step is to construct a full weighted graph for the input document with the formula $G = (V, E, W)$. Here, V indicates the collection of vertices and E denotes the word vector and each vertex corresponds to a phrase. W refers to the set of weights that are assigned to graph G's edges. The weight w is assigned to the edge (u,v) of equation E using the following logic. Assume $S(u) = \{W_1u, W_2u, \ldots, W_su\}$ and $S(v) = \{W_1v, W_2v, \ldots, W_sv\}$ are the sentences that describes to the vertex u and v, respectively. For each word in the sentence, inverse sentence frequency (isf) and term frequency (tf) have been taken into account.

$$isf(w, D) = \log \frac{D}{\{S \in D : w \in S\}}. \quad (1)$$

As defined by Eq. (1), $isf(w, D)$ is the word w inverse sentence frequency in the input document D. The number of sentences whereby the word w seems is represented by $\{S \in D : w \in S\}$ and the number of sentences found in the input document is denoted as D.

$$W(u,v) = \frac{\sum_{\forall x \in S(u) \cup S(v)} ((S(u))\, tf_x(S(v)))(isf_x(D))^2}{\sqrt{\sum_{\forall y \in S(u)} (tf_y(S(u))(isf_y(D))^2} \times \sqrt{\sum_{\forall Z \in S(v)} (tf_Z(S(v))(isf_Z(D))^2}}. \quad (2)$$

Now, every pair of words is compared to one another using the *isf*-modified-cosine similarity, which is employed in Eq. (2) as the weight for the edge (u, v). Conventional cosine similarity merely takes into account the relative occurrence of the words that are comparable in the text document; in this instance, all sentence vectors should have the same dimension. In contrast, the *isf*-modified-cosine similarity measure considers both the various sentence lengths in the text document and the various degrees of significance for the relevant words in the sentences.

By eliminating the edges whose weights are below the threshold value (t), which corresponds to the weight of the graph's total weighted edges, the outcome is a whole weighted graph that has been made sparse. This graph displays the similarity graph for the original document. Every sentence that corresponds to a node on the graph G receives a score for the purpose of weight assignment. In this method, each node's mean weight of the edges occurring to a text document determines its Text Rank score, emphasizing the edges (E) corresponding weights, as opposed to conventional PageRank, where each node's initial PageRank value is set to $1T$, where T represents the graph's total node count.

$$TR(S(u)) = \frac{d}{T} + (1-d)\sum_{v \in adj(u)} \frac{TR(S(v))}{\deg(S(v))}. \quad (3)$$

Where,

Text rank $TR(S(u))$ indicates that sentence S relates to node u, whereas text rank $TR(S(v))$ indicates that sentence S related to node v includes $v \in adj(u)$, when a node v corresponds to a sentence S, its degree is $\deg(S(v))$, all of the nodes found in the graph is represented as T and d is a "damping factor." Finally for summarization, using the sentence index as a guide, the best n scored sentences are chosen and rearranged and the n output sentences construct the summary. For overcoming the OOV issues, the sentences above the word frequency length are considered for the abstractive BLG pointer generator algorithm.

BLG Pointer Generator Algorithm

Sentences from the document are separated based on the threshold value, then the separated sentences are given as the input to the BLG pointer generator algorithm [17]. This network allows for both word generation from a predefined vocabulary and word copying by pointing and also hybrid the network by an attentional sequence to a pointer network-based sequence baseline.

Figure 2 illustrates the proposed abstractive BLG pointer approach for text summarization. After embedding the word using the pretrained model (i.e. bidirectional encoder representation transformer), the article tokens W are given one at a time to the encoder (for a single-layer bidirectional GRU) [18], generating an encoder sequence of hidden state h. The single layer unidirectional LSTM decoder [19] enters the decoder state S after obtaining the word embedding of the preceding word at step t. The prior word is located in the reference summary during training, and the decoder emits it during testing. At, the attention distribution is computed.

$$e_i^t = v^T \tanh(W_h h_i + W_s S_t + b_{\text{attn}}), \quad (4)$$

$$a^t = \text{softmax}(e^t). \quad (5)$$

Where, v, W_h, W_s and b_{attn} are learnable parameters. The attention distribution, which instructs the decoder wherever to focus in order to produce the next word, is able to be regarded as the probability distribution across the original words. The context vector h_t^*, which is a weighted sum of the encoder hidden states, is generated using the attention distribution:

$$h_t^* = \sum_i a_i^t h_i. \quad (6)$$

In order to create the vocabulary distribution P_{vocab}, in this stage the information collected from the source is combined using the decoder state S and passed via two linear layers, which can be thought of as a fixed-size description represented by h_t^*.

$$P_{\text{vocab}} = \text{softmax}\left(V'\left(V\left[S_t, h_t^*\right] + b\right)b'\right). \quad (7)$$

Where learnable parameters are represented as $V'\left(V\left[S_t, h_t^*\right] + b\right)b'$, P_{vocab} defines a probability distribution of the vocabulary's words. The context vector h_t^* and attention distribution a^t in the pointer-generator model are computed using equations (6) and (7). Furthermore, from the decoder state S_t, context vector h_t^* and the decoder input S_t, determining the probability of generation P_{gen} belongs to [0, 1] for timestep t as

$$P_{\text{gen}} = \sigma\left(w_{h^*}^T h_t^* + w_x^T x_t + w_s^T S_t + b_{ptr}\right). \quad (8)$$

Where learnable parameters are represented as $w_{h^*}^T$, w_s^T, w_x^T vectors and scalar b_{ptr} and sigmoid function is denoted as σ. P_{gen} is then employed as a soft switch for selection between creating a word using the vocabulary by taking a sample from P_{vocab}, and using a sample from the attention distribution at to copy a word using the input sequence. Let the expanded vocabulary for every document represent the combination of every word found in the original document and the vocabulary.

$$P(w) = P_{\text{gen}} P_{\text{vocab}}(w) + (1 - P_{\text{gen}}) \sum i : w_i = w^{a_i^t}. \quad (9)$$

It should be noted that $P_{\text{vocab}}(w)$ is zero when w is an OOV word; and the $\sum i : w_i = w^{a_i^t}$ variable is 0 when w is not found in the source document. The main benefits of pointer-generator models is the ability to generate OOV words; while models like baseline are limited to their predefined vocabulary. The coverage strategy maintains a coverage vector c^t that represents the total of the attention distributions over every prior

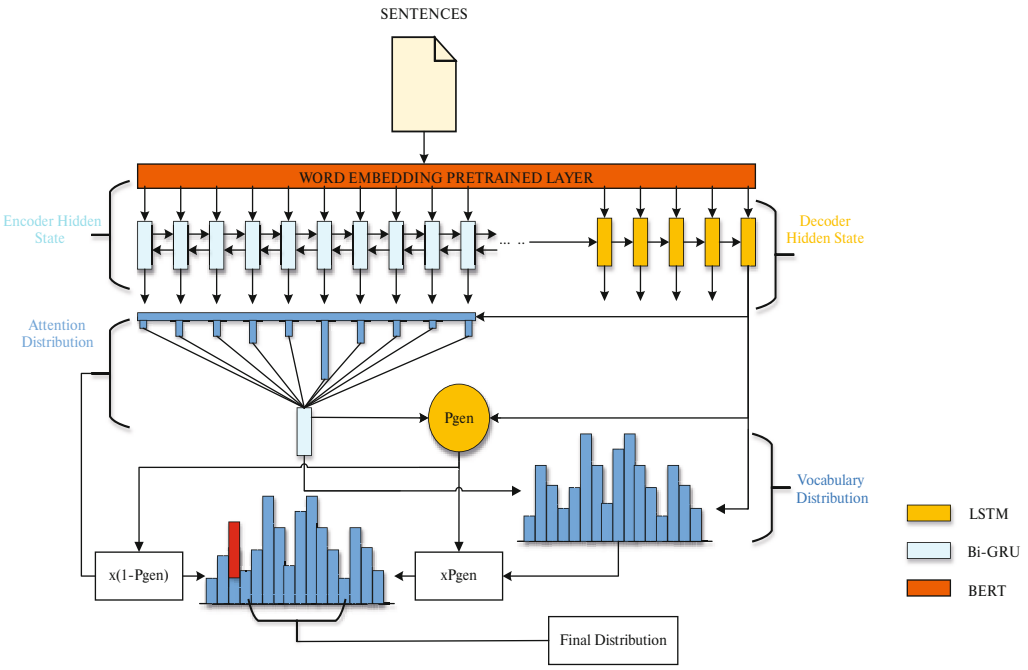

Fig. 2. Architecture of the proposed Abstractive BLG pointer approach.

decoder timesteps in order to tackle the repetition problem:

$$c^t = \sum_{t'}^{t-1} a^{t'}. \tag{10}$$

Intuitively, c^t is a (unnormalized) distribution over the source document words that represents the degree of coverage that those words have received from the attention mechanism so far. The coverage vector is added as additional input to the attention process to:

$$e_i^t = v^T \tanh\left(W_h h_i + W_s S_t + W_c c_i^t + b_{\text{attn}}\right). \tag{11}$$

Where W_h is a vector of learnable parameters that has the same length as v. This guarantees that the decision made by the attention mechanism, which is deciding where to focus next, is based on a review of its past decisions, which are summarized in c^t. This ought to facilitate the attention mechanism's ability to avoid from continuously focusing on the same areas, which should prevent repetitive text from being produced.

Word Embedding Pre-trained Layer

A pre-trained BERT [20] model is used in the architecture for illustrating the input words depending on the pointer-generator network. This technique presents each word in the decoder and encoder layers as an integrated vector before it is passed to the subsequent levels.

$$x_t = g(w). \tag{12}$$

Where function g converts every word in the input into the associated embedding vector within the model that has already been trained. It leads to an improvement in the attention distribution by improving the network ability to the inputs determine hidden states due to every time step t are fully and accurately measured. Furthermore, because the P_{gen} value is an input component that has been fully learned, the calculation of the value will be more accurate.

$$P_{\text{gen}} = \sigma\left(w_{h*}^T h_t^* + w_s^T S_t + w_x^T g(w) + b_{ptr}\right). \tag{13}$$

The word embeddings trained by a different pre-trained model, rather than their word indices in the vocabulary of the original text, are now the input. In both the network's encoder and decoder, the new embedding layer is used. Every decoder input will be embedded before being given to it at each time step, whereas the embedding from the source text is fed into the encoder as input for time steps. This proves that each word has a unique representation in the network in both the source text and the summary.

Squid Game Optimizer for TextR-BLG Algorithm

Squid game, also known as ojingeo, is a variation of tag and hopscotch that originated as a playground activity for Korean kids [21]. It is a multiplayer game with two main objectives: either the teams must defeat one another or the attackers must succeed in their attack. Although participants can play in huge, open, indoor, or outdoor spaces, large, sandy fields are where the squid game is frequently played. Regarding

the size and proportions of the playground, there are currently no suggestions or rules. Based on the squid game's past, the squid-shaped playfield appears to be about half the size of a regular basketball court. The procedures for choosing the LSTM and Bi-GRU's parameters in the best possible way.

Step 1: Initialization

The parameters of the LSTM and Bi-GRU, such as neurons per hidden layer, learning rate, batch size, and activation function are regarded as parameters in LSTM and Bi-GRU. These parameters are considered as attributes $X1$, $X2$, $X3$... Xn.

$$F = (X1, X2, X3 \ldots Xn.). \quad (14)$$

Step 2: Fitness Function

K-fold cross validation is used for evaluating the parameters of the LSTM and Bi-GRU. These characteristics are separated into K roughly equal-sized subsets at random. One fold is used as the validation set at a time, while the remaining folds $(K-1)$ are used as the training set. Accuracy is used to calculate the average test error and is obtained using an equation.

$$\text{fitness} = \{\text{maximize}(acc_{cv})), \quad (15)$$

$$acc_{cv} = \frac{1}{n} \sum_{v_i, y_i \in D} \delta\left(I(D \backslash D_{(i)}, v_i), y_i.\right. \quad (16)$$

Maximising Accuracy using the aforementioned equation serves as the basis for evaluating the fitness function. The dataset D is randomly divided into k roughly equal-sized, mutually exclusive subsets (the folds), $D_1, D_2, \ldots, D_k$ in k-fold cross-validation, also known as rotation estimation. The inducer is trained on D_t and $D \backslash D_t$ tested on k times, with each time being $t \in \{1, 2, \ldots, k\}$. The total number of instances in the dataset serves as the cross-validation estimate of accuracy. Formally, define $D_{(i)}$ as the test set including an instance $x_i = v_i, y_i$, and the cross-validation accuracy estimate that follows.

Step 3: Updating

After the process of the Fitness function, each and every other set of attributes is updated using equation (17). The updating equation is determined from the squid optimization algorithm.

$$X_i^{offNew3} = X_i^{Sccoff} + r_1 BS - r_2 X_k^{SccDef} \begin{cases} i = 1, 2 \ldots o. \\ k = 1, 2 \ldots p. \end{cases} \quad (17)$$

Step 4: Termination

Parameters for the LSTM and Bi-GRU are updated using the above equation (16) until the maximized accuracy is attained for the selected parameters. After attaining the parameters of the LSTM and Bi-GRU the entire process will be terminated.

Algorithm 1. Hybrid TextR-BLG generator algorithm

Input: Size of summary = n (say) and Text extracted from the target document D

Articles = $A1, A2, \ldots AN$ // Collected articles from PubMed dataset
T = Combine (Articles) //
S = Split_Sentences (T)
WF = Word_Freq_Len (S)
If ($L \leq 30$)
$S1$ = Sentences below or equal to length 30
Do extractive approach
else
$S2$ = Sentences above word length 30
Do Abstractive approach
#Abstractive approach
PB = BERT ($S2$)
BE = Bi-GRU encoder (PB)
AD = Attention Distribution (BE)
L = LSTM decoder ($S2$)
P_{gen} = AD + L
xP_{gen} = Vocabulary Distribution + AD
$x(1 - P_{gen})$ = P_{gen} + Attention Distribution
Final Distribution = $xP_{gen} + x(1 - P_{gen})$
#Extractive approach
Sim_matrix = Similarity matrix ($S1$+ Final Distribution)
GP = Graph plot (Sim_matrix)
SR = Sentence Ranking (GP)
SUM = SUMMARY_GEN (SR)

Output: Summary of the input document.

RESULT AND DISCUSSION

The proposed automatic summarization has implemented by utilizing the hybrid text rank and BLG pointed generator algorithm. This model is implemented on PYTHON 3.8 software with the system configuration of Intel Core i7 CPU, NVIDIA GeForce RTX 3070 GPU and 64GB RAM. The automatic long document summarization model is based on the collected PubMed dataset. These articles collected are combined together and the sentences are separated from the paragraph. The word frequency length of the sentences are considered based on the threshold value. The word frequency length with equal to or greater than 30 is considered as abstractive approach. These sentences are reframed using the BLG pointed generator algorithm and the sentences are combined with the sentences with the word frequency length of below or equal to 30. The combined sentences are considered for the abstractive approach, where the sentences are considered for attaining the similarity matrix and based on the similarity matrix,

Table 1. ROUGE score for the proposed model TextR-BLG pointer algorithm

TextR-BLG pointer Algorithm	ROUGE-1		
	Average_R	**Average_P**	**Average_F**
	0.53	0.62	0.68
	ROUGE-2		
	0.45	0.65	0.60
	ROUGE-L		
	0.56	0.66	0.64

the graph is plotted. From this graph plot, the sentences are ranked and based on the damping ratio the summary is generated.

Dataset

Dataset [22] for summarization of long documents is done by using the PubMed dataset that contains paper id, article and abstract for the medical related topic. The dataset used in this model is split into training, testing and validation. Total 1,33,215 number of instances are used for training, testing and validation. 1,19,924 number of instances are used for training the Bi-GRU encoders and LSTM decoders for training the dataset to reframe the sentences for summarization.

Table 1 illustrates the average recall, F1-score and precision for ROUGE-1, 2, and L. ROUGE-N is an n-gram recall between a candidate summary and a set of reference summaries. Performance metrics for text summarization techniques such as the proposed TextR-BGL and the existing approaches such as TF-TDF, Text Rank Score (TRS), Pegasus and Seq2Seq model are compared for evaluating the model performance. The performance of the proposed and existing model is evaluated based on the ROUGE-1, ROUGE-2, ROUGE-L, Meteor Score, BERT Score and Bleu Score.

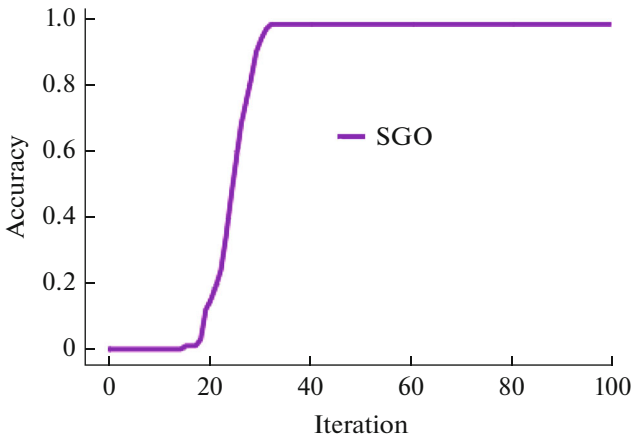

Fig. 3. Convergence plot for Squid game optimizer.

Figure 3 illustrates the convergence plot for Squid game optimizer to optimally select the number of hidden layers, the number of LSTM units in each layer, the learning rate, the batch size and the dropout rate. An illustration of how the optimization has changed over time in relation to the number of individuals assessed is referred to as a convergence plot. The optimal solution to the optimization problem can be determined by examining the convergence history graph.

Figure 4 illustrates the comparison of the proposed and existing summarization approach. In Recall-Oriented Understudy for Gisting Evaluation (ROUGE)-1, the system and reference summary's overlap in terms of unigrams (per word) is analyzed. The extractive and abstractive approach used for comparing the proposed TextR-BGL are Pegasus, Seq2Seq, TF-IDF and TRS. The attained ROUGE-1 score for both proposed and existing model are 0.59, 0.46, 0.43, 0.39 and 0.31. From these values, the summarization of the proposed TextR-BGL attains a better score than the existing models. A comparison of ROUGE-2 is illustrated in Figure 5. ROUGE-2 is the bigram overlap between the reference summaries and the framework. The attained score of ROUGE-2 for the proposed and existing model are 0.58, 0.44, 0.41, 0.36 and 0.28 for TextR-BGL, Pegasus, Seq2Seq, TF-IDF and TRS.

ROUGE-L comparison for the proposed and existing summarization approach is illustrated in Fig. 6. The term "ROUGE-L" describes statistics based on the Longest Common Subsequence (LCS). The LCS issue automatically determines the longest co-occurring sequence of n-grams and considers sentence-level structure similarities. The extractive and abstractive approach used for comparing the proposed TextR-BGL are Pegasus, Seq2Seq, TF-IDF and TRS. The attained ROUGE-L values for both proposed and existing models are 0.62, 0.49, 0.45, 0.32 and 0.307. Figure 7 illustrates the meteor score comparison for the proposed and existing summarization approach. METEOR (Metric for Evaluation of Translation with Explicit ORdering) is a metric used to evaluate machine translation output. On the basis of the harmonic mean of unigram recall and precision, the metric gives recall a higher weight than accuracy. In addition to the standard exact word matching, it provides several extra features (such as synonymy and stemming matching) that are absent from other metrics. The attained METEOR values for both proposed and existing model are 0.67, 0.59, 0.53, 0.48, 0.41 for the proposed TextR-BGL, Pegasus, Seq2Seq, TF-IDF and TRS.

Figure 8 illustrates the BERT score for the proposed and existing summarization approach such as TextR-BGL, Pegasus, Seq2Seq, TF-IDF and TRS. An automated evaluation tool called BERTScore is used to evaluate the manner in which text generating

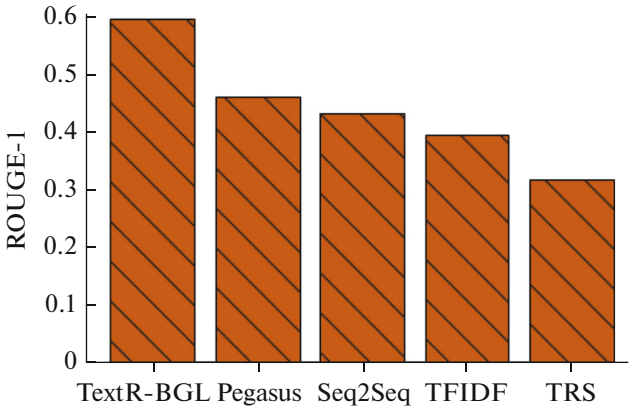

Fig. 4. ROUGE-1 comparison for proposed and existing summarization approach.

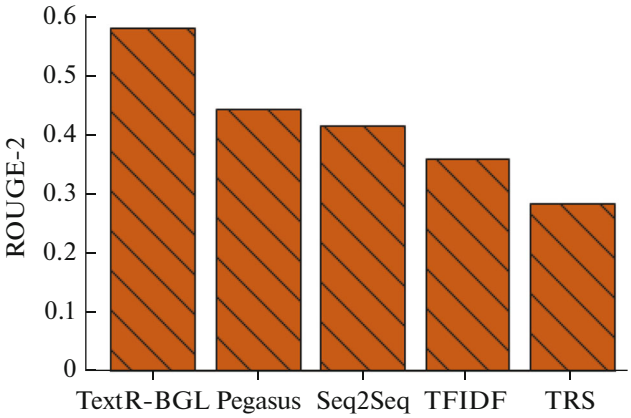

Fig. 5. ROUGE-2 comparison for proposed and existing summarization approach.

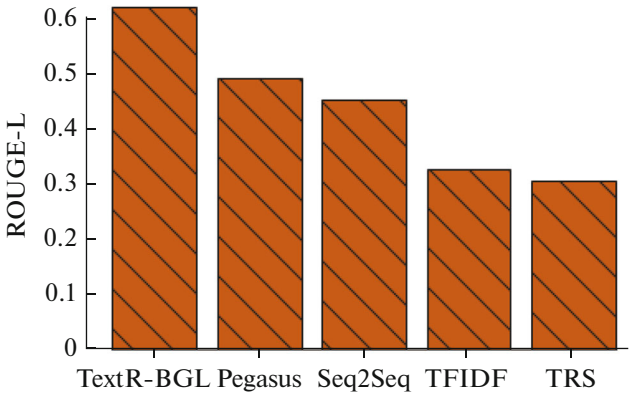

Fig. 6. ROUGE-L comparison for proposed and existing summarization approach.

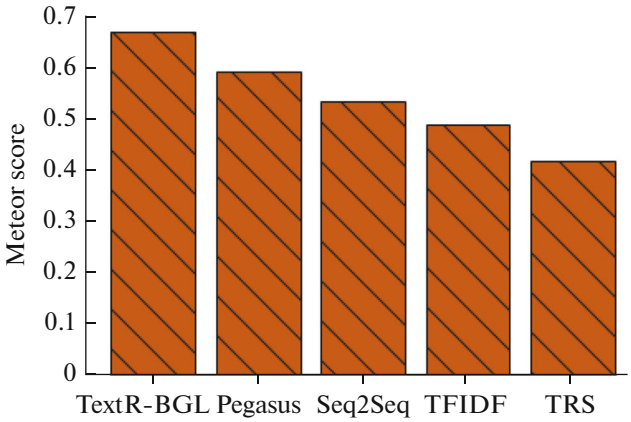

Fig. 7. Meteor Score comparison for proposed and existing summarization approach.

algorithms work. Instead of focusing on computing token-level syntactical similarities like other prominent techniques, BERTScore calculates the semantic similarity between reference and hypothesis tokens. The attained values for TextR-BGL, Pegasus, Seq2Seq, TF-IDF and TRS are 0.92, 0.86, 0.73, 0.68 and 0.61. BERTScore for the proposed model is better than the existing model. BlueScore attained for the proposed and existing models are graphically represented in Fig. 9. A machine translation algorithm called BLEU (bilingual evaluation understudy) is used to assess the accuracy of content that has been converted between natural languages. The degree to which a machine's output resembles that of a person is known as its quality. The most well-liked automated and affordable metric is still BLEU, which was the first to assert a significant correlation with human quality determinations. The attained BLEU score for the proposed and existing models such as TextR-BGL, Pegasus, Seq2Seq, TF-IDF and TRS are 0.78, 0.72, 0.66, 0.59 and 0.53. From the evaluation, the proposed TextR-BGL model attains better performance metrics than the existing model. Hence, the proposed model summarizes the long document with better meaning than the existing model.

Table 2 demonstrates the comparison of automatic text summarization for the PubMed dataset. The comparison between the proposed approach and existing methods are PEGASUS, BigBird PEGASUS, DANCER PEGASUS, Sent-CLF, ExtSum-LG+, Lead-10M, SumBasic, LexRank, Seq2Seq and Pointer-gen. Compared to the proposed automatic text summarization, the proposed model ROGUE values are greater than the existing model. Thus, the designed hybrid approach for automatic text summarization by handling out-of-vocabulary words using the TextR-BLG pointer algorithm performs better than the existing extractive and abstractive approach.

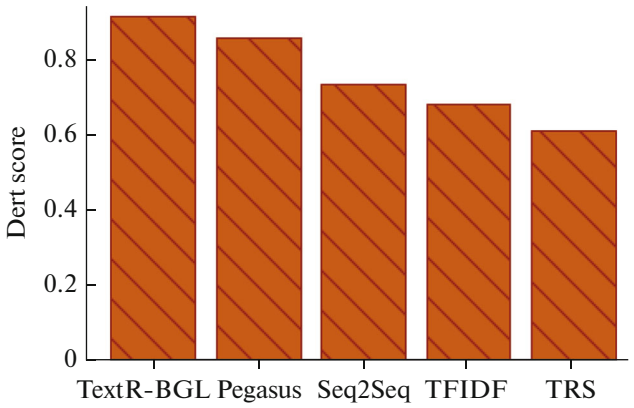

Fig. 8. BERT Score comparison for proposed and existing summarization approach.

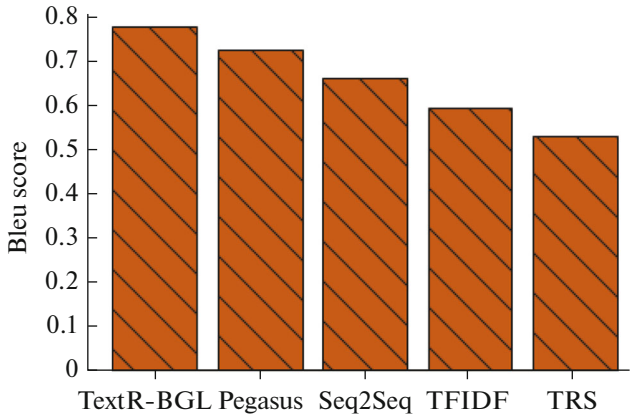

Fig. 9. Bleu Score comparison for proposed and existing summarization approach.

CONCLUSIONS

Thus, the proposed hybrid approach for automatic text summarization using the TextR-BLG pointer algorithm is done and validated with performance metrics. Text summarization is the most significant process in today's environment due to the production of large amount of data from the net source as well as to reduce the reader's time. Due to their effectiveness and fluency, extractive techniques are frequently utilized in text summary. However, the majority of extractive models in use today hardly ever catch intersentence links, especially in lengthy documents. They frequently ignore the impact of timely information on collecting significant information. In this methodology, a lengthy document is provided as an input for artificial text summarization based on a hybrid extractive and abstractive technique. The document's sentences are evaluated for word frequency length and divided into two processing approaches, such as the extractive and abstractive approaches, depending on the threshold values. This model's text algorithm determines the sentence's similarity score and validates its results using the plotted graph. The sentences that are longer than the threshold level are considered as abstractive approach. For learning the meaning from the sentences, the Optimized BERT-LSTM-BiGRU (BLG) based generator algorithm is utilized. Finally, the BLG generating algorithm's reframed sentence is taken into account in order to calculate the similarity score and create the graph. For the purpose of creating the summary, the sentences are prioritized according to the plotted graph. For evaluating the performance of the algorithm, the ROUGE score, Meteor score, BLEU score and BERT score. The attained performance values of the proposed model are 59.2, 58.4, 62.3, 0.67, 0.92 and 0.78 for ROUGE 1, ROUGE 2, ROUGE L, Meteor score, BERT score and BLEU score. Thus, the evaluated values of the proposed hybrid TextR-BLG algorithm performs better than the existing model. In the future

Table 2. Comparison of automatic text summarization techniques for PubMed dataset

MODEL	METRICS		
	ROUGE-1	ROUGE-2	ROUGE-L
PEGASUS	45.49	19.90	42.42
BigBird PEGASUS	46.32	20.65	42.33
DANCER PEGASUS	46.34	19.97	42.42
Sent-CLF	45.01	19.91	41.16
ExtSum-LG+	45.30	20.42	40.95
Lead-10	37.45	14.19	34.07
SumBasic	37.15	11.36	33.43
LexRank	39.19	13.89	34.59
Seq2Seq	31.55	8.52	27.38
Pointer-gen	35.86	10.22	29.69
Proposed TextR-BGL	59.2	58.4	62.3

scope, the proposed text summarization algorithm can be used in many online book reading apps for summarizing the main important content of the story to highlight the topic of the story in short verse.

AUTHOR CONTRIBUTION

The corresponding author claims the major contribution of the paper including formulation, analysis and editing. The co-author provides guidance to verify the analysis result and manuscript editing.

FUNDING

The authors declare that no funds, grants, or other support were received during the preparation of this manuscript.

ETHICS APPROVAL AND CONSENT TO PARTICIPATE

This article is a completely original work of its authors; it has not been published before and will not be sent to other publications until the journal's editorial board decides not to accept it for publication.

CONFLICT OF INTEREST

The authors of this work declare that they have no conflicts of interest.

REFERENCES

1. Cui, P., Hu, L., and Liu, Yu., Enhancing extractive text summarization with topic-aware graph neural networks, *Proc. 28th Int. Conf. on Computational Linguistics*, Barcelona, 2020, Scott, D., Bel, N., and Zong, Ch., Eds., Int. Committee on Computational Linguistics, 2020, pp. 5360–5371. https://doi.org/10.18653/v1/2020.coling-main.468.
2. Givchi, A., Ramezani, R., and Baraani-Dastjerdi, A., Graph-based abstractive biomedical text summarization, *J. Biomed. Inf.*, 2022, vol. 132, p. 104099. https://doi.org/10.1016/j.jbi.2022.104099
3. Karthick, S. and Muthukumaran, N. Deep regression network for single-image super-resolution based on down- and upsampling with RCA blocks, *Natl. Acad. Sci. Lett.*, 2023. https://doi.org/10.1007/s40009-023-01353-5
4. Rouane, O., Belhadef, H., and Bouakkaz, M., Word embedding-based biomedical text summarization, *Emerging Trends in Intelligent Computing and Informatics. IRICT 2019*, Saeed, F., Mohammed, F., and Gazem, N., Eds., Advances in Intelligent Systems and Computing, vol. 1073, Cham: Springer, 2020, pp. 288–297. https://doi.org/10.1007/978-3-030-33582-3_28
5. Bian, J., Huang, X., Zhou, H., and Zhu, Sh., GoSum: Extractive summarization of long documents by reinforcement learning and graph organized discourse state, *arXiv Preprint*, 2022. https://doi.org/10.48550/arXiv.2211.10247
6. Song, S., Huang, H., and Ruan, T., Abstractive text summarization using LSTM-CNN based deep learning, *Multimedia Tools Appl.*, 2019, vol. 78, no. 1, pp. 857–875. https://doi.org/10.1007/s11042-018-5749-3
7. Gupta, S., Sharaff, A., and Kumar Nagwani, N., Graph ranked clustering based biomedical text summarization using top k similarity, *Comput. Syst. Sci. Eng.*, 2023, vol. 45, no. 3, pp. 2333–2349. https://doi.org/10.32604/csse.2023.030385
8. He, P., Peng, B., Wang, S., Liu, Ya., Xu, R., Hassan, H., Shi, Yu., Zhu, C., Xiong, W., Zeng, M., Gao, J., and Huang, X., Z-Code++: A pre-trained language model optimized for abstractive summarization, *Proc. 61st Annu. Meeting of the Association for Computational Linguistics*, Toronto, 2023, Rogers, A., Boyd-Graber, J., and Okazaki, N., Eds., Association for Computational Linguistics, 2023, vol. 1, pp. 5095–5112. https://doi.org/10.18653/v1/2023.acl-long.279
9. Rani, R. and Lobiyal, D.K., An extractive text summarization approach using tagged-LDA based topic modeling, *Multimedia Tools Appl.*, 2021, vol. 80, no. 3, pp. 3275–3305. https://doi.org/10.1007/s11042-020-09549-3
10. Zhang, Yo., Li, D., Wang, Yu., Fang, Ya., and Xiao, W., Abstract text summarization with a convolutional seq2seq model, *Appl. Sci.*, 2019, vol. 9, no. 8, p. 1665. https://doi.org/10.3390/app9081665
11. Dou, Z.-Yi., Liu, P., Hayashi, H., Jiang, Z., and Neubig, G., GSum: A general framework for guided neural abstractive summarization, *Proc. 2021 Conf. of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, Toutanova, K., Rumshisky, A., Zettlemoyer, L., et al., Eds., Association for Computational Linguistics, 2020, pp. 4830–4842. https://doi.org/10.18653/v1/2021.naacl-main.384
12. Liu, Ya., Fine-tune BERT for extractive summarization, *arXiv Preprint*, 2019. https://doi.org/10.48550/arXiv.1903.10318
13. Xu, J., Gan, Z., Cheng, Yu., and Liu, J., Discourse-aware neural extractive text summarization, *Proc. 58th Annu. Meeting of the Association for Computational Linguistics*, Jurafsky, D., Chai, J., Schluter, N., and Tetreault, J., Eds., Association for Computational Linguistics, 2019, pp. 5021–5031. https://doi.org/10.18653/v1/2020.acl-main.451
14. Alguliyev, R.M., Aliguliyev, R.M., Isazade, N.R., Abdi, A., and Idris, N., COSUM: Text summarization based on clustering and optimization, *Expert Syst.*, 2019, vol. 36, no. 1. https://doi.org/10.1111/exsy.12340
15. Zhang, S., Celikyilmaz, A., Gao, J., and Bansal, M., EmailSum: Abstractive email thread summarization, *Proc. 59th Annu. Meeting of the Association for Computational Linguistics and the 11th Int. Joint Conf. on Natural Language Processing*, Zong, Ch., Xia, F., Li, W., and Navigli, R., Eds., Association for Computational Linguistics, 2021, vol. 1, pp. 6895–6909. https://doi.org/10.18653/v1/2021.acl-long.537

16. Xiong, C., Li, X., Li, Yu., and Liu, G., Multi-documents summarization based on textrank and its application in online argumentation platform, *Int. J. Data Warehousing Min.*, 2018, vol. 14, no. 3, pp. 69–89. https://doi.org/10.4018/ijdwm.2018070104
17. See, A., Liu, P.J., and Manning, C.D., Get to the point: Summarization with pointer-generator networks, *Proc. 55th Annu. Meeting of the Association for Computational Linguistics*, Vancouver, Canada, 2017, Barzilay, R. and Kan, M.-Ye., Eds., Association for Computational Linguistics, 2017, vol. 1, pp. 1073–1083. https://doi.org/10.18653/v1/p17-1099
18. Suleiman, D. and Awajan, A., Deep learning based abstractive text summarization: Approaches, datasets, evaluation measures, and challenges, *Math. Probl. Eng.*, 2020, vol. 2020, p. 9365340. https://doi.org/10.1155/2020/9365340
19. Gambhir, M. and Gupta, V., Deep learning-based extractive text summarization with word-level attention mechanism, *Multimedia Tools Appl.*, 2022, vol. 81, no. 15, pp. 20829–20852. https://doi.org/10.1007/s11042-022-12729-y
20. Wang, Q., Liu, P., Zhu, Z., Yin, H., Zhang, Q., and Zhang, L., A text abstraction summary model based on BERT word embedding and reinforcement learning, *Appl. Sci.*, 2019, vol. 9, no. 21, p. 4701. https://doi.org/10.3390/app9214701
21. Azizi, M., Shishehgarkhaneh, M.B., Basiri, M., and Moehler, R.C., Squid game optimizer (SGO): A novel metaheuristic algorithm, *Sci. Rep.*, 2023, vol. 13, no. 1, p. 5373. https://doi.org/10.1038/s41598-023-32465-z
22. Albertvillanova, Huggingface, 2021. https://huggingface.co/datasets/ccdv/pubmed-summarization/tree/main. Cited October 17, 2023.