Multimedia Tools and Applications (2024) 83:18987–19013
https://doi.org/10.1007/s11042-023-16199-8

Graph-based extractive text summarization based on single document

Avaneesh Kumar Yadav[1] · **Ranvijay**[1] · **Rama Shankar Yadav**[1] · **Ashish Kumar Maurya**[1]

Received: 21 September 2022 / Revised: 14 June 2023 / Accepted: 4 July 2023 /
Published online: 25 July 2023

Abstract
Day by day, the amount of online and offline text data is growing tremendously from various sources like legal documents, medical documents, news articles, etc. Manual text summarization of large documents is unfeasible and costly because it takes much time and requires more effort. As a consequence, various graph-based text summarization techniques have been designed which provide thoroughly and well-prepared summaries of documents. The problems issues that exist in these techniques are redundancy of data, loss of information and readability. To overcome these problems, we have proposed a textual graph-based extractive text summarization technique called TGETS, for extracting essential information from a single document. In the proposed approach, a graph's node is denoted as group of sentences in the document and an edge of the graph is represented as an association between two sentences. The summary generation is based on the sum of sentence weight and the average weight of the textual graph. The performance of proposed approach is evaluated on the BBC news articles dataset through the ROUGE-metric (R_1 and R_2). The proposed approach in the range of 100-200 words length summary offers better scores of 19.88%, 38.76%, and 30.73% for R_1 under precision, recall and F_1-score with respect to the existing PageRank (PR) method. Similarly, for R_2, the proposed approach exceeds by 32%, 26.99%, and 29.01% for precision, recall, and F_1-score with respect to existing PageRank (PR) method.

Keyword Text summarization · Extractive text summarization · Lemmatization · Textual graph · ROUGE

✉ Avaneesh Kumar Yadav
avaneesh17@mnnit.ac.in

Ranvijay
ranvijay@mnnit.ac.in

Rama Shankar Yadav
rsy@mnnit.ac.in

Ashish Kumar Maurya
ashishmaurya@mnnit.ac.in

1 Department of Computer Science & Engineering, Motilal Nehru National Institute of Technology Allahabad, Prayagraj 211004, India

1 Introduction

Nowadays, the growth of textual data information in digital format is huge on the website and almost 80% of text data information is available on the Internet. Due to overloaded textual information over Internet, several indispensable issues and challenging tasks arise. These issues include summarization of multiple languages of text documents, single document summarization based on extractive text summarization, positioning of sentence order of the summary, loss of information [16], quality of summary, and computationally expensive [8, 25]. On the way to resolve these issues, there are several challenges, such as to build an extractive text summarization method for huge text corpus (documents) like news articles, novels, and stories [40, 53], to address relevant keywords from text documents based on summarizer tools. Another challenge while summarizing a lengthy document is the compression ratio [5]. The efficiency of a system in terms of computation cost is affected by the length of a text document. Currently, researchers and academicians find it challenging to create an effective extractive summary for long text documents. Several researchers and academics are now working on text corpus in fields such as text categorization, information retrieval, document summary, sentiment analysis [5], question answering [8], i.e., Chatbot, mining large data, social media analysis, radio news, scientific papers, encyclopedias, technical reports, journal articles, web pages, etc. and text summarization is classified into several categories, that is shown in Fig. 1.

1. Generality can be classified into query and generic-based summarization (shown in Fig. 1). Query-based summarization has a collection of homogenous documents dealt by multi-documents. It is brought out from the corpus for query results [42]. Query-generated summaries always focus on query-related content. Generic summarization is extracting relevant information from one or more input documents, providing it to the common sense of its content [1, 16]. In comparison, generic-based cannot depend on user query but query-based summarization depend on the user query.
2. Purpose can be categorized into a general and specific domain i.e., shown in Fig. 1. Both domains are independent of the text summarization document. There are different types of domains used in a particular field, such as sports documents, legal documents, medical documents, news articles, etc.
3. Input-type documents are divided into single and multi-document, as shown in Fig. 1. A single document provides single input and then generates a single output summary but a multi-document provides a multi-input at the time of summarization and generates a single output summary. In existing work, several authors have used single or multi-document types such as [29, 42, 45].

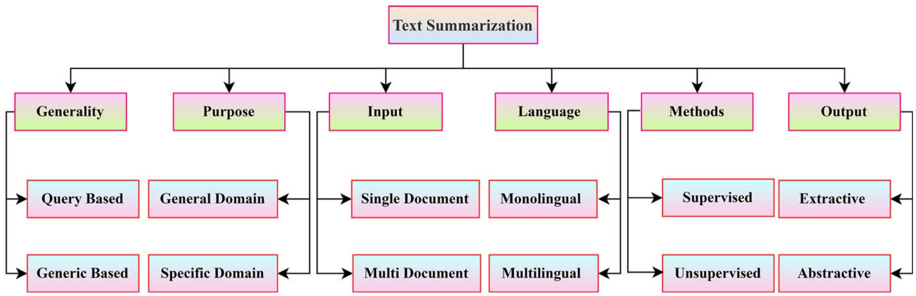


Fig. 1 Classification of text summarization approaches

4. Languages are separated into two groups: monolingual and multilingual, i.e., also shown in Fig. 1. When both the source and target documents are in the same language, a summarization system is known as monolingual language. If the source text is written in multiple languages (for example, Hindi, English, Nepali and Urdu), and the summary is produced in each of those languages, the text summarization system is called multilingual.
5. Methods for text summarization can be divided into two categories: supervised and unsupervised, i.e., also shown in Fig. 1. Both are possible depending on whether or not training data is necessary. A supervised approach is trained using labeled data to pick out concepts from text documents. The most common method is unsupervised, and it produces summaries from text documents. They do not rely on any training examples with labels applied by users.
6. Output is classified into extractive text summarization and abstractive text summarization, as shown in Fig. 1. Extractive text summarization generates an extractive summary from the given text documents. Extractive text summarization focuses on significant sentences from a given text document with many approaches such as machine learning, linguistics, statistics, optimization approach [24], etc. Extractive text summarization has a conspicuous rank assigned to sentences in text documents. Conspicuous sentences are selected on the rank of sentences previously chosen to produce the original documents' summary. Abstractive text summarization creates an abstract overview containing phrases and words in the original document. Besides this, the overview generated by abstractive text summarization also includes words that are not there in the document but takes those words from the dictionary to generate a meaningful summary. Abstractive text summarization has a more challenging task because it has semantic restrictions and language dependency due to paraphrasing. Abstractive text summarization can be applied in relatively small documents for the most straightforward tasks like sentence fusion, sentence compression [21, 24, 40], and generation of keywords, headlines, or titles [35, 37]. However, abstractive text summarization is more complicated than extractive text summarization. The extractive text summarization raises the achievability to attain the quality of document summarization. Extractive text summarization and abstractive text summarization, when combined, they are known as a hybrid approach. The extractive text summarization approach is dissimilar to the abstractive text summarization approach. Several researchers have focused their literature study on extractive text summarization summarization approaches. In this article, the proposed approach use extractive text summarization techniques.

In this paper, we focus on the extractive text summary based on textual graph-based extractive text summarization (TGETS) approach. The textual graph is created from a single documents. The graph's vertices represent the sentences of the text document, and edges represent the common words between two sentences. Several approaches have opted for the extractive text summarization (ETS) process. Subsequently, the Latent Semantic Analysis (LSA) [12, 19, 47], PageRank [11], TextRank and LexRank [13, 30], Graph-based approach [11, 12], Hidden Markov Models [52], and frequency-based term weight approaches [1, 16] are used in ETS summarization. ROUGE evaluation metrics compare two summaries (such as reference and system-generated summary) to identify the best possible summary. Sentences with the most common words are considered related and given higher priority. Similar sentences are deemed redundant, and one of them is ignored while preparing the summary. The objective of this research is to summarize the large news article of different domain such as sport, business, entertainment, technical, politics in very concise and informative way.

Summarization reduces the time for accessing and understanding the large documents. To achieve this objective, the following contribution are given below.

- We find the keyword from each sentence of single document with the help of pre-processing tool. After that, the root words are generated in sentences by lemmatization process and these root words help to generate textual graph of the single document.
- We propose an approach for computation of common words between two sentences and assign weight to the edges accordingly. Also, the computing score for each sentence (node of the textual graph) is formalized afterwards.
- We propose a method for computing average weight of the textual graph of the single document. This average weight is then used to generated summary by elimination sentences from the document.
- We calculate the result of summarizing the document by using ROUGE-metric, and results of state-of-the-art (i.e., TextRank (TR), KUSH (KH), LexRank (LR), and PageRank (PR)) summary are compared with the proposed approach of the experimental results.

The rest of this paper is structured as follows: Section 2 deals with related works of ETS. Section 3 gives a proposed textual graph-based extractive text summarization (TGETS) approach. Section 4 discuss experimental setup and result analysis based on experiments. At last concludes the paper and highlights future directions.

2 Related work

The most frequent technique for automated ETS summarization is ranking sentences or phrases and generating summaries. The majority of the current approaches have used sentence ranking. Graph, phrase, and word ranking are the three types of ranking methodologies. Ranks are calculated using sentence ranking algorithms. These algorithms use the relevant sentence as their aide to create ranks. Frequent words are given higher priority. Also, objects, proper nouns, and locations are given higher priority. The formal qualities of the phrase (underlined, quotation, italicized, typography, emphasis, emboldened) are considered in text ranking algorithms. Furthermore, sentences beginning with phrases like “As a result,” “Finally,” and “Briefly” is defined as assigned phrases throughout the text and the subsequent statements in these sentences are designated as crucial sentences. Similarly, the title of the content (text) to be summarized is used to evaluate it. Sentences with terms from the title are deemed to be added to the summary. The proportions of sentences are also considered by sentence ranking algorithms, with more significant sentences being given more weight. Sentences are given weightage based on their position and whether or not they contain numerical values. Elbarougy et al. [11] described an Arabic-specific ETS summarization method. The authors presented a graph-based document representation model with sentences as vertices. The Modified PageRank (MPR) method is used, with an initial score of the number of nouns in this sentence given to each node. The cosine similarity of two words is used to calculate the weight of each edge between them. The suggested method has three primary steps: pre-processing, feature extraction, and graph formation, followed by applying the new MPR algorithm and creating a relevant summary. Roul [41] suggested an ETS method for the Hindi language. They tagged lexical and meaningful information from Hindi novels and stories utilizing topic modeling and LDA. There is a lack of corpus and processing tools for Hindi, authors created their own corpus and tools. Experiments conducted result in the best outcomes when compared to baseline and definitive topic modeling methods. The authors (Nandhini and Balasundaram [36]) detailed the creation and evaluation of an extractive summary strategy to assist learners with reading challenges. Text analysis algorithms commonly use graph-based representations because they offer very effective solutions. The authors presented TextRank (Mihalcea and Tarau [30, 31]), which includes a graph-based model for ETS summarization based on text document intersections. LexRank, a node centrality approach based on eigenvector centrality, was introduced by Erkan and Radev [13]. The PageRank (PR) (Brin and Page [7]) technique, introduced to obtain central sentences in a document utilizing mutual information between sentence sets and words, inspired both the LexRank (LR) and TextRank (TR) methods. Parveen et al. [38] used the LexRank and TextRank methods. Uçkan and Karcı [49] proposed KUSH pre-processing tools and maximum independent sets (MIS). KUSH (KH) tool performs closed-meaning words which differ from other alternative search words. MIS has removed the vertices from the textual graph to generate a relevant summary.

Canhasi and Kononenko [8] performed query-focused multi-document summarization (Q-MDS) using weighted Archetypal Analysis, a matrix factorization technique. The artificial data set was displayed as an undirected sentence similarity network, where nodes represented sentences, edges represented similarity between connected nodes, and each phrase was related to the given query. Salton et al. [43] used link creation for automated document summary and characterized texts with graphs. They defined the structure by analyzing the documents' text relationships and compared the summary to reference generated summary by human's. Medelyan [28] proposed a graph-based method to provide semantic continuity, with nodes representing document concepts and edges indicating semantic linkages between them. The longest and shortest pathways are described as the strongest and weakest links, respectively after a graph diameter computation is conducted for all nodes in the graph. Local similarities were used to create nodes and edges, whereas graph topologies and documents were produced by Chen et al. [9]. Although most authors have concentrated on ETS, some have succeeded with abstractive summarization too. The RNN structure was used to do abstractive multi-sentence summarization in Nallapati et al. [35]. Similarly, Moawad and Aref [32] did abstractive summarizing works. Belwal et al. [6] discussed the similarity between nodes of the textual graph and the weight of graph edges. Authors calculated the rank of sentences through TextRank algorithms, and these ranks are kept in descending order. The compression ratio (CR) generates a relevant summary from the input documents taken from two datasets, Opinosis (total 51 documents) and CNN/DailyMail News (related to technical articles, social, and political). Azadani et al. [4] discussed graph-based biomedical text summarization using sentence clustering and itemset mining. Authors generated a meaningful summary and evaluated the result of the summary by ROUGE metric from the huge number of datasets. A brief summary of the literature analysis is also provided in Table 1.

On the study of the above-related work, following are the research gaps related to text summarization.

1. For single-document text summarization, there has been few research that uses a graph-based approach and relatively few works have tried to use edge-weighted method to determine the most appropriate relevant sentences of the text document.
2. In the previous work, redundancy still exists in the summarized document. This leads to reduce the quality of summary.
3. The earlier developed method does not use the benefit of common-word edge weight to find the most significant sentence.
4. The task of cutting down on document processing and summarization time has yet to receive much focus.
5. Most of the existing work take more time even for a small summary generation.

Table 1 Literature Survey of ETS approaches

Authors, Year	Model/Approach	ToD (S/M)	ToS	Datasets
Canhasi and Kononenko [8]	Matrix Factorization and Graph-based Methods	M	Query-Focused	DUC-2005, DUC-2006
Rani and Lobiyal [40]	LDA tool	S	Extractive	Hindi novels (Munshi Premchand's)
Nasar et al. [37]	Itemset and graph Clustering	S	Extractive	DUC-2002, DUC-2004
Nallapati et al. [35]	Recurrent Neural Network	M	Abstractive	CNN/DailyMail, DUC-2002
El-Kassas et al. [12]	Statistical Approach	S	Extractive	DUC-2001, DUC-2002
Erkan and Radev [13]	Eigen Centrality and Graph-based Methods	M	Extractive	DUC-2004
Roul [41]	PageRank	S, M	Extractive	DUC-2002, DUC-2006
Nandhini and Balasundaram [36]	Machine Learning	S	Extractive	Educational Data
Parveen et al. [38]	ILP and Graph-based Methods	S	Extractive	DUC-2002
Uçkan and Karcı [49]	KUSH Tool/Eigen Vector Centrality	S	Extractive	DUC-2002
Salton et al. [43]	HLGA	M	Abstractive	Encyclopedia Article
Medelyan [28]	Lexical Chain and Graph-based Method	S	Key phrases Extraction	Offline Document Data
Chen et al. [9]	Random Walk and Graph-based Method	S	Key Term Extraction	Mandarin and the Sinica Taiwan English corpus
Moawad and Aref [32]	Semantic Graph Reducing	S	Abstractive	None
Belwal et al. [6]	Topic Modeling	S	Extractive	CNN/DailyMail, Opinosis
Azadani et al. [4]	TGraph Clustering & Frequent Itemset/TextRank	S	Extractive	Biomedical articles
Tomer and Kumar [48]	Genetic Algorithm (GA)	M	Extractive	DUC-2002, DUC-2003, DUC-2004
Awan and Beg [3]	Top-Rank and Graph-based Methods	S	Key-phrases Extraction	KDD
Mutlu et al. [34]	Machine Learning	S	Extractive	SIGIR -2018
Al-Sabahi et al. [2]	Hierarchical Structured, Self-Attentive Model	S	Extractive	CNN/DailyMail, DUC-2002
Fang et al. [14]	Graph-based Method	S	Extractive	News Article and DUC-2002
Fattah and Ren [15]	FFNN, PNN, MR, GMM, and GA-based models	M	Extractive	DUC-2001

ToD Type of Document, *ToS* Type of Summarization, *S* Single document, *M* Multi-document

Section 3 provide more detail about the textual graph-based extractive text summarization approach.

3 Proposed textual graph-based extractive text summarization (TGETS) approach

We propose a textual graph-based extractive text summarization based on a single document summarization methodology for extracting meaningful summary from the input document. There are three key phases to the proposed document summarization approach such as pre-processing, processing, and post-processing, i.e., shown in Fig. 2. In the first phase, pre-processing concerns with paragraph segmentation, sentence segmentation, word tokenization, and elimination of stop-word. After that finding root words based on the lemmatization process. In the second phase, there are several sub-steps, such as creation of textual graph, the common word between two sentences forms the basis of connectivity between the sentences (i.e., edges of the textual graph), and the computation sum of weight of each sentences S_i, and compute the average weight of textual graph (G_T). In last post-processing phase, the selection of sentences based on the average weight of textual graph followed by sentence order. It generates a summary based on the top N phrases at this stage of the proposed technique, and N-word (viz., 100-words, 150-words, 200-words, etc.) summaries are prepared independently.

3.1 Text Pre-processing

The vast majority of dense datasets are unconfigured, and just a tiny percentage of essential datasets are structured. Structured datasets may be stated in a table's rows and columns or use several tags. Ignoring complex and time-consuming processes in the current investigation made it possible to provide a specific structure to the data. Some pre-processing is required

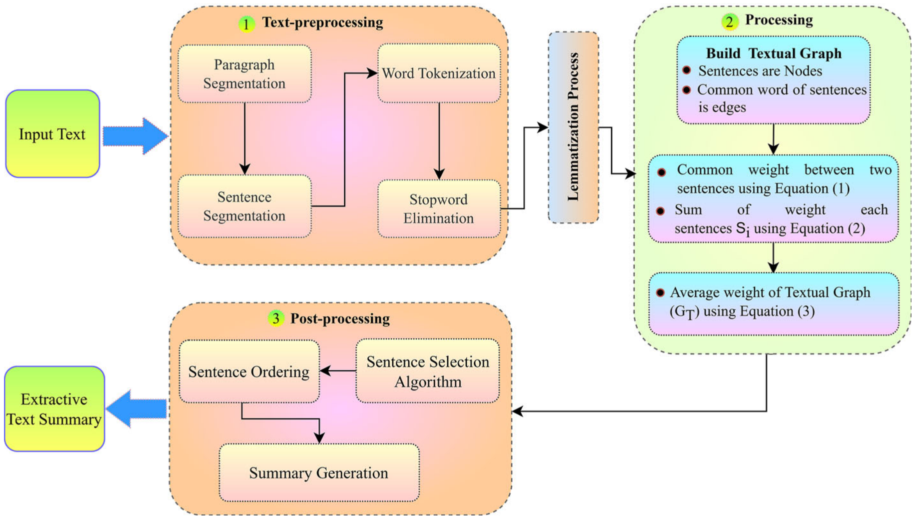


Fig. 2 Architecture of text processing for ETS summarization

for datasets that lack specified integrity yet must be structured. Since dense clusters exist in natural language, it is projected that translating texts to useable formats and isolating them from non-discriminatory data will grow the system's performance. The data to be analyzed in text summarization procedures must be pre-processed. Stop words (conjunctions, pronouns, prepositions) are expressions with no identifying properties and should be deleted before summarizing the dataset. As a result, non-representative terms inside sentences are eliminated from the original datasets, so text summarization can be done by converting the data to a suitable format for analysis. As a result, the processing load is reduced, but certain words are still utilized. Text can be conveyed with graphs and is saved in files with the .txt extension. The BBC News article datasets are utilized in this study. The normalization procedures in this study are carried out using the python language library and tags, which include removing spaces, stop words, and unnecessary characters, i.e., shown in Fig. 2. The data pre-processing and preparation stage begins with these normalization steps.

The pre-processing methods include paragraph segmentation, sentence segmentation, tokenization of words, and stop-word elimination. Most of these strategies are often utilized in an ETS system's pre-processing phase [26, 45, 50, 53].

Paragraph segmentation In general, an input text corpus is a collection of paragraphs or sections separated by the user. Each paragraph has its own set of main points, and a varied summary is built by merging the most relevant components from each paragraph [40].

Sentence segmentation It separates the sentences of a paragraph into sentences [16]. In many circumstances, utilizing end markers like ".", "?"or "!" to split sentences isn't appropriate. Words like "e.g.,", "i.e.,", "31.5", "Mr.", "Dr.", or "etc." lead to the incorrect detection of sentence borders if we rely on these markers. Regular expressions and basic heuristics address this problem [18, 24].

Word tokenization It separates the text into individual words. White space, a dot, a dash, a comma, and other symbols are used to divide words [18]. Example: "Humans are at risk from Covid-19 virus." like ["Humans", "are", "at", "risk", "from", "Covid-19", "virus"], these sentences have a total of seven tokens [46].

Stopword elimination Stopwords include auxiliary verbs, prepositions, pronouns, articles, and determiners. They have been left out because they do not add anything to the analysis [20] and have no bearing on how the essential sentences are chosen [16, 46]. In English languages, almost 179 stopwords are there, such as ["he", "she", "our", "am", "they", "where", "have", "can", "could", "may", etc.]. In the above example, three stopwords ["are," "at," "from"] are available; hence, they are eliminated so that sentence behaves as "Human's risk Covid-19 virus".

3.2 Process of root-word conversion using lemmatization method

It is the process of putting together the inflected elements of a word so that it can be identified as a single element, known as the word's lemma or meaningful vocabulary form. This is similar to stemming but gives meaning to specific words [23]. It combines content with similar meanings to a single word in simple terms [39], as shown as Fig. 3.

Example: (1) Form= "studies" → Meaningful-information = 3^{rd} person → singular number → present tense of the verb study → lemma word = "study".

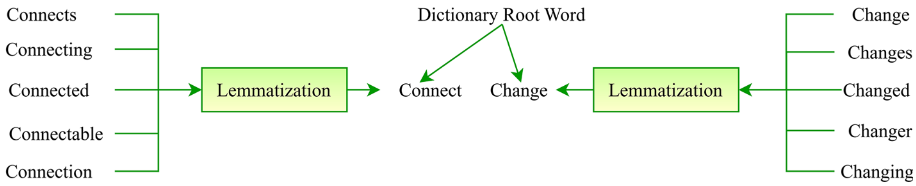


Fig. 3 Representation of keywords through lemmatization process

(2) Form= "studying" → Morphological-information = Gerund of verb study → Lemma = "study".

Lemmatization is the process of converting numerous keywords into dictionary root-words. Table 2 represent several root-words and keywords similar to Fig. 3. These type of root-words are used for connecting two vertices (i.e., two sentences) via a common term that is more useful during the textual graph construction.

Flow chart of lemmatization process According to the flow chart of lemmatization, we can convert the input words into root-words using the database of dictionary words. Its conversion is based on dictionary rules and then provides actual root-words. Lemmatization process always produces meaningful root-words. In this paper, the graph creates an association of two nodes through the common root-word of the two nodes (i.e., two sentences). The conversion of root-words by lemmatization process is shown in Fig. 4 and Algorithm 1.

Furthermore, when building graphs of closely related words that differ in spelling but are semantically derived from the similar root-words are treated as separate words. This makes it more laborious to discern interconnections and associations between the sentences.

3.3 Creation of weighted textual graph

In this subsection, a textual graph has been created; the textual graph presents several ways of representing information through weighted and non-directional graphs. The data undergoes pre-processing steps before being used to create the textual graphs. The proposed document summarization model constructs textual graph after data pre-processing and lemmatization. This procedure is shown in the "text processing and building of textual graph" stages of Fig. 2. Figure 5(a) and (b) represent a textual graph for a single text document. The textual

Table 2 Other lemmatization examples of root words and keywords

Root-word	Keywords
Study	Study, Studies, Studying, and Studied
Trouble	Trouble, Troubling, Troubled, and Troubles
Principle	Principle, Principalities, Principality, and Principles
Affect	Affect, Affection, Affected, Affecting, Affectation, and Affects
Change	Change, Changeable, Changes, Changing, Changed, and Changer
Connect	Connect, Connection, Connecting, Connected, Connects, Connectable, and Connector
Multiple	Multiplier, Multiply, Multiplied, Multiplying, Multipliable, Multiplies, Multiplicate, Multiplication

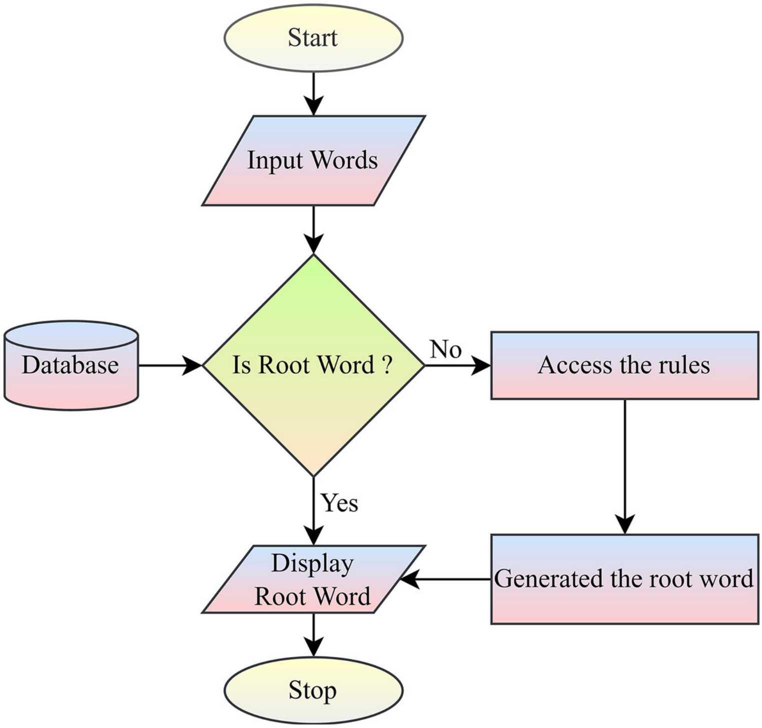


Fig. 4 Systematic diagram of lemmatization process for root-words

graph can be represented as G_T=(V, E), where $V \in (v_1, v_2, v_3, \ldots, v_{n-1}, v_n)$, and $E \in (e_1, e_2, e_3, \ldots, e_{n-1}, e_n)$. It is a weighted undirected graph, i.e., calculating the intersection of words in between sentence with all other sentences reveals the relationship levels of the sentence pairs S_i and S_j is the same as between S_j and S_i. The weight of edges has been decided by common words between sentences using (1). The sentences are represented as vertices of the textual graph. The vertex/node values in the set (S) on the textual graph (G_T) are connected in an undirected manner. Graphs are stored in an adjacency matrix, where the size of the adjacency matrix is $N \times N$. The values of the adjacency matrix are the weight of edges in between two sentences. Complete textual graph are treated as dense matrix whose time complexity is O(N^2). Otherwise, the textual graph is a sparse matrix. In the sparse matrix are a smaller number of edges than dense matrix, where the time complexity of textual graph is O(V+E) [10, 22, 51], where vertices (V) is the total number of sentences (S=N) such that E=N then O(N+N)≈ O(2N)≃ O(N). Therefore, worst and best time complexity for the document are O(N^2) and O(N), respectively.

(a) **Compute common weight between two sentences in the textual graph**
Calculate the common weight between S_i and S_j. Its edges are determined by the intersection ($\bigcap$) of the sentences S_i and S_j. This weight reflects the thickness of the graph edges or their weight. Figure 5(b) illustrates the circular weight graph is used to represent the weighted values of the thickness graph. A general mathematical formula is mentioned in (1) and also detail discuss in Algorithm 2.

$$Weight\ of\ e\left(S_i, S_j\right) = set\ of\ keywords\ (S_i) \bigcap set\ of\ keywords\ (S_j) \quad (1)$$

Algorithm 1 Find the root keywords using lemmatization techniques

Input: Provide input clean sentence (i.e., keywords of sentences) from the text documents D
Output: Display root keywords

```
begin
    Dictionary-based rule= [prefix, suffix, infix];
    Vocabulary = [100000 root-words] /* At least 100000 root-words are
        available in the Database file                                */
    Input vocabulary = check input vocabulary (Input keywords, vocabulary)
    if input vocabulary == actual keywords then
        return root word;
    else
        for rule in rules do
            x = find root − word (Input keywords, rule);
            Input vocabulary= check input vocabulary (x, vocabulary);
            if input vocabulary == actual keywords then
                return root-word
                break;
            else
                continue;
            end
            root-word = root-word;
            return root-word
        end
        root-word = root-word;
        return root-word
    end
    return root keywords
end
```

Algorithm 2 Computation of edge weight between two sentence in Textual graph (G_T))

Input: Set of keywords for each nodes S_i in Graph 'G', where i=1 to n, and using Algorithm 1.
Output: Weight of each edges e(S_i, S_j)

```
begin
    Initilize weight of each edges, e(S_i, S_j)=0;
    for i=1 to n do
        for j=i+1 to n do
            if i not equal j then
                Compute e(S_i, S_j) using (1);
            else
                e(S_i, S_j)=0;
            end
        end
    end
    return edge weight, e(S_i, S_j)
end
```

where $\bigcap$ represent the common word between two sentences are called as edge weight of textual graph.

(b) **Weight of each sentence S_i in textual graphs**
Consider the text graph's weight in relation to the node (the document's sentences). The following textual graph (from Fig. 6) illustrates the specifics of (2) and also, it is discussed

through Algorithm 3.

$$W(S_i) = \sum_{j=i+1}^{n} S_{ij} \tag{2}$$

Algorithm 3 Computation of weight of each sentence in Textual Graph (G_T)

Input: Weight of edges in Graph 'G', using Algorithm 2.
Output: Weight of each node S_i, (W(S_i)).

begin
  Initilize each node weight W(S_i)=0, where i=1, 2, ..., n;
  for *i=1 to n* **do**
    for *j=i+1 to n* **do**
      Compute weight of each sentences S_i (W(S_i)) using (2);
    end
  end
  return W(S_i)
end

(c) **Average weight of Textual Graph (G_T)**

Determine the score (weight) of each sentence in the text document to determine the average weight of the textual graph. It requires an integer value for the textual graph's average weight. Once the values are maximized and equal to the sentences' weight values, the mathematical condition is represented in (3).

$$AW(G_T) = \left\lceil \frac{\sum_{i=1}^{n} W(\mathrm{S}_i)}{N} \right\rceil \tag{3}$$

Algorithm 4 Average weight of Textual Graph (G_T)

Input: Weight of each node W(S_i), where i=1, 2, ..., n; using Algorithm 3.
Output: Average Weight of textual graph, (AW(G_T)).

begin
  Initilize weighted_sum=0;
  for *i=1 to n* **do**
    weighted_sum=weighted_sum+$W(S_i)$;
  end
  AW(G_T)=$\left\lceil \frac{weighted\_sum}{N} \right\rceil$
  return AW(G_T)
end

when nodes (sentences) of the textual graph (G_T) weight are less than the average weight of the graph (G_T). So that we pick the better sentence from the text document for the creation summary. Its mathematical condition is shown in (4) and it is described detail in Algorithm 4.

$$W(S_i) < AW(G_T) \tag{4}$$

Algorithm 5 is performed for textual graph-based extractive text summarization (TGETS). The proposed TGETS steps are described in Algorithm 5.

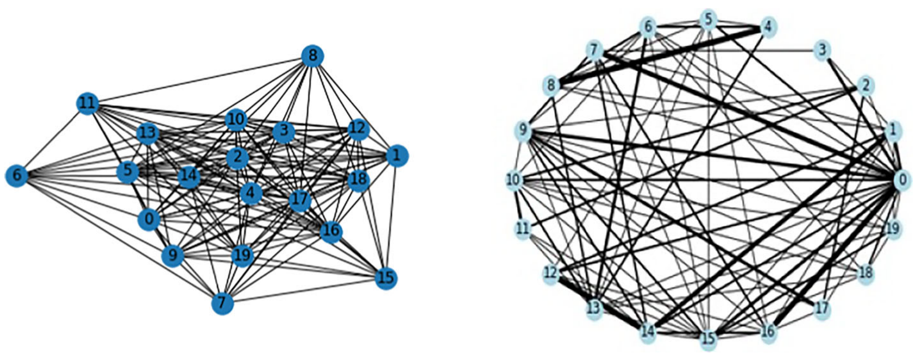
Algorithm 5 Proposed algorithm of textual graph-based extractive text summarization (TGETS)

Input: Main keywords of text Document, D
Output: Extractive text summary
begin
  Find the set of keywords in each sentence (S_i) in Document, D; //call Algorithm 1
  Find the common keywords between each sentence $S_{i,j}$, where $i \in 1, 2, ..., n$ and $j \in 1, 2, ..., n$ for the computing edge weight between $S_{i,j}$, //call Algorithm 2;
  Compute the weight of each sentence S_i, //call Algorithm 3;
  Compute the average weight of the document, //call Algorithms 2 and 4;
  All sentences S_i whose weight is less than the average weight of the document, discard sentence from the document;
  Generate a summary of document D_i by arranging all the sentences whose score is greater than the average weight of the document;
  Summary generated based on the number of words, such as 100-, 150-, 200-, 250-, 300-, and 350-word length;
  return Extractive text summary
end
terminated

3.4 Post-processing of text document

After the computation of the average weight of textual graph of each node, the next step is to generate the summary. The user decides the summary length in terms of number of words (X) (like 100-word, 200-word, 300-word, etc.), i.e., the number of words between document size and length of the summary generator. The number of sentences constituting the summary vary depending on the length of the individual sentence. A sentence having more rank is treated as the more important sentence in constituting the summary. Thus, we sort the nodes (sentences represented as nodes) in descending order of ranks, depending upon the limitation of summary length (decided by the user). We extract up the top-X number of words in document sentences and arrange them in the proper sentence order of the document, i.e., a quality summary of the single document.

For better sake of understanding of the above algorithm, it has been illustrated by an example. A textual graph G_T = (V, E) consists of a set of vertices (V) and edges (E) [55],



(a) Random generated graph (b) Circular generated graph

Fig. 5 Graphs (**a**) and (**b**) generated from the same input document

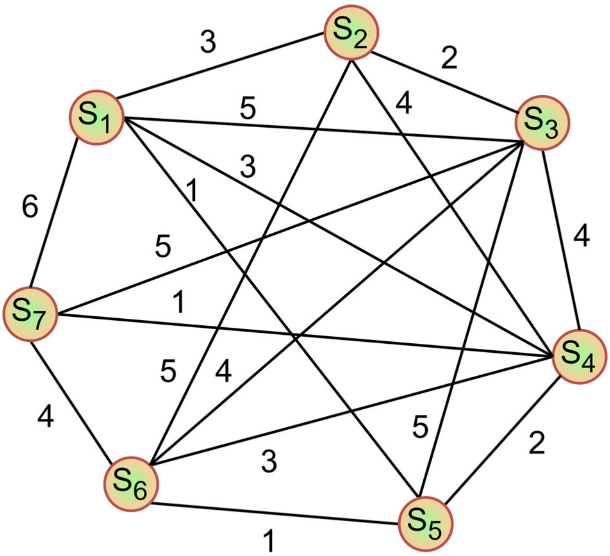


Fig. 6 Representation of weighted Textual Graph (G_T (V, E))

where the set of vertices of the graph is sentences of a document and edges (common word) in between two sentences. Figure 6 shows the document's 7-sentences ($S_1,\ S_2,\ \ldots,\ S_7$), and the weight of the edge between two sentences are common words.

The common words between two sentences are calculated by (1), i.e., the weight of edges. There are seven sentences ($S_1,\ S_2,\ \ldots,\ S_7$), where each sentence's weight is calculated based on weight of edges between two sentences which is shown in Fig. 6.

$$
\begin{aligned}
W(S_1) &= S_1S_5 + S_1S_4 + S_1S_3 + S_1S_2 + S_1S_7 = 1 + 3 + 5 + 3 + 6 = 18 \\
W(S_2) &= S_2S_1 + S_2S_3 + S_2S_4 + S_2S_6 = 3 + 2 + 4 + 5 = 14 \\
W(S_3) &= S_3S_1 + S_3S_2 + S_3S_4 + S_3S_5 + S_3S_6 + S_3S_7 = 5 + 2 + 4 + 5 + 4 + 5 = 25 \\
W(S_4) &= S_4S_1 + S_4S_2 + S_4S_3 + S_4S_5 + S_4S_6 + S_4S_7 = 3 + 4 + 4 + 2 + 3 + 1 = 17 \\
W(S_5) &= S_5S_1 + S_5S_3 + S_5S_4 + S_5S_6 = 1 + 5 + 2 + 1 = 9 \\
W(S_6) &= S_6S_2 + S_6S_3 + S_6S_4 + S_6S_5 + S_6S_5 + S_6S_7 = 5 + 4 + 3 + 1 + 4 = 17 \\
W(S_7) &= S_7S_1 + S_7S_3 + S_7S_4 + S_7S_6 = 6 + 5 + 1 + 4 = 16
\end{aligned}
$$

Calculate the average weight of the textual graph (from Fig. 6)

$$
\begin{aligned}
AW(G_T) &= \left\lceil \frac{W(\mathcal{S}_1) + W(\mathcal{S}_2) + W(\mathcal{S}_3) + W(\mathcal{S}_4) + W(\mathcal{S}_5) + W(\mathcal{S}_6) + W(\mathcal{S}_7)}{7} \right\rceil \\
AW(G_T) &= \left\lceil \frac{18 + 14 + 25 + 17 + 9 + 17 + 16}{7} \right\rceil \\
AW(G_T) &= \left\lceil \frac{116}{7} \right\rceil = \lceil 16.571429 \rceil \\
AW(G_T) &= 17
\end{aligned}
$$

Now, we can eliminate the less important sentences from the text document based on an average weight of sentences and their weight of sentences using (4), such as,

$$W(S_2),\ W(S_5),\ W(S_7) < W(S_1),\ W(S_3),\ W(S_4),\ W(S_6)$$

According to the example, the values of $W(S_2),\ W(S_5),\ W(S_7)$ are lower than $AW(G_T)$. It is eliminated from the text document since it does not meet the criterion for the average weight of the textual graph. Other weight of sentences are $W(S_1),\ W(S_3),\ W(S_4),\ W(S_6)$ then user generates a summary from the important sentences-based weight of sentences $S_1,\ S_3,\ S_4,\ S_6$. These weights of sentences are put into descending order of weight sentences. The user-definable summary is based on the number of words summarized from the remaining sentences after they hold the criterion of the average weight of G_T to generate a relevant summary of the text document.

4 Experimental setup and result analysis

The detail experimental setup is performed on the computer with the processor (an Intel (R) $Core^{(TM)}$ i5-1035G1 10^{th} Gen.) and CPU (Central Processing Unit) @ 1.19 GHz, 8.00 GB RAM size, Ubuntu 21.04 using Jupyter Notebook (Python3.6). The texts in the BBC news article dataset are saved in .csv files. The Python language library and tags is used to perform the normalization procedures, including removing unwanted characters, whitespace, expressions, and non-discriminatory words known as stop words. Also, we discuss details of dataset in next subsection.

4.1 Description of dataset

The BBC-article dataset is used in two major sections; one is reference article and other one is its corresponding reference summary. In the BBC-article dataset, there are multiple articles on different domain with their respective reference summaries. News articles are given in different contexts, including entertainment, sport, business, etc. The details about the BBC-article dataset are listed as part of Table 3.

This paper summarizes the BBC news articles' document using index data value on ten positions of entertainment articles', with 25 sentences and 533 words used to summarize the documents.

Table 3 The description of BBC-article dataset

Dataset Name	Deatils	Brief description of datasets	
BBC-article (total 2225 articles)	Domain-Name	Reference Article	Reference Summary
	Entertainment articles	386	386
	Sport articles	511	511
	Politics articles	417	417
	Business articles	510	510
	Technical articles	401	401

4.2 Evaluation performance metric

The ROUGE metric is regarded as the gold standard for measuring the quality of generated summaries with the reference summary. Here, both summaries are matched, and the score is calculated based on common content in both summaries. The match content refers to a number of uni-gram matching and bi-gram matched. These matching are termed as $ROUGE_1$ and $ROUGE_2$, respectively [33, 40, 44–46, 53, 54]. Now on the basis of matching, we compute in terms of precision, recall, and F_1-score for $ROUGE_1$ and $ROUGE_2$ independently. Equation (5) is used to determine the ROUGE-metric for various granularities and discuss the following ROUGE-metric below [17, 50].

$ROUGE_N$ (R_N): R_N is based on N-gram comparison of a reference and candidate summary, where N=1, 2.

$ROUGE_1$ (R_1): R_1 is based on uni-gram comparison of a reference and candidate summary.

$ROUGE_2$ (R_2): R_2 is based on a bi-gram comparison of a reference and candidate summary.

$$ROUGE_N = \frac{\sum_{S\in(Ref\,Summ)} \sum_{N-gram\in(S)} Count_{match}(N-gram)}{\sum_{S\in(Ref\,Summ)} \sum_{N-gram\in(S)} Count\ (N-gram)} \tag{5}$$

$$ROUGE_L = \frac{LCS(P,\ Q)}{n} \tag{6}$$

where N is the $gram_n$ length, and $Count_{match}$ (N-gram) is count of several numbers of n-grams present in the reference and candidate summary. $ROUGE_N$ is evaluated by (5). In this case, the denominator equals the sum of the count of n-grams in the reference summary. Similarly, The $ROUGE_L$ value also determines the longest common word sequence in the two sub-word sequences P and Q. Two-word sequences, P (i.e., reference summary length is n) and Q (i.e., candidate summary length is m) are supplied. Equations (7), (8), and (9) are used to calculate the $ROUGE_L$ value of the sequences [1, 24, 33, 46, 50, 54].

$$R_{LCS} = \frac{LCS(P,\ Q)}{n} \tag{7}$$

$$P_{LCS} = \frac{LCS(P,\ Q)}{m} \tag{8}$$

$$F_{LCS} = \frac{(1+\gamma^2).R_{LCS}.P_{LCS}}{\gamma^2.R_{LCS}+P_{LCS}} \tag{9}$$

where, LCS(P, Q) is the length of a longest common subsequence of P and Q, and γ is the ratio of P_{LCS} and R_{LCS}. In the next sub-section, we discuss about the results obtained and compare with the state-of-the-art methods.

4.3 Experimental setting

In this section, we discuss experimental work with state-of-the-art methods and perform the analysis in detail in the form of tabular as well as graphical ways. In the Appendix section, the input single text document and reference summary are given in Appendix A and B covers system generated summary of the given document on the number of words length provided by users. This has been done to show as an example for a better understanding of the proposed approach.

4.4 Described various methods

Here, we discuss various methods (such as LexRank (LR), TextRank (TR), PageRank (PR) and KUSH (KH)) used in the experimental section, which are discussed below in detail with mathematical formulas.

LexRank (LR) "The strength of the similarity link can be used to calculate LexRank. It generates the similarity network directly from the cosine values and frequently ends up with a considerably denser yet weighted graph. It has a stochastic matrix and normalizes the row sums of the appropriate transition matrix."

For weighted or unweighted graphs, the resulting equation is a modified version of LexRank and represented in (10).

$$L\left(V_i\right) = \left[\left(\frac{d}{N}\right) + (1-d) * \sum_{Vj \in adj(V_i)} \frac{W(V_i, V_j)}{\sum_{k \in adj[V_j]} W(k, V_j)} L(V_j)\right] \tag{10}$$

where $L\left(V_i\right)$ = LexRank value of the sentence V_i , N = the total number of sentences (nodes) in the graph, d = empirically determined damping factor but it lies $d \in [0...1]$. by default, d=0.85, $adj(V_i)$ = set of the sentences that are neighbors of V_i in the graph, and $W(V_i,\ V_j)$ = the weight of the link bewteen sentence V_j and V_i.

TextRank (TR) "TextRank is a graph-based algorithm for text summarization and keyword extraction. It ranks words or phrases based on connections and patterns of co-occurrence within a text. The algorithm ranks the tokens in the text according to their scores, highlighting the most crucial ones for summary or keyword detection [30]."

The vertices (V_i) are connected to a set of vertices (V_j). Here, V_i and V_j are sentences. Then the TextRank of vertices, $T(V_i)$ is shown in (11) [27, 31].

$$T\left(V_i\right) = \left[(1-d) + d * \sum_{j \in V_i} \frac{T(V_j)}{V_j}\right] \tag{11}$$

PageRank (PR) "The PageRank value for a page V_i is calculated by dividing the PageRank values for each page V_j in the set V_i (all pages linked to the page V_i) by the number $|V_j|$ of outbound links from the page V_j [7]".

The vertices (V_i) are connected to a set of vertices (V_j). Then the PageRank of vertices (V_i) is shown in (12).

$$P\left(V_i\right) = \left[\left(\frac{1-d}{N}\right) + d * \sum_{j \in n}^{n} \frac{P(V_j)}{|V_j|}\right] \tag{12}$$

KUSH (KH) Based on Uçkan and Karcı [49] concept that terms relating to the nodes in the independent set should be eliminated from the summary. The nodes forming the Independent Sets (IS) on the graphs were determined and erased from the graph. As a result, the evaluation of the summary was constrained before the nodes' impact on the global graph. Because of this

Table 4 100-word length text summary (BBC News Article)

ROUGE Metric		TGETS	TR	KH	LR	PR
R_1	Precision	0.98275	0.57857	0.89915	0.75164	0.79523
	Recall	0.81428	0.78641	0.76428	0.58362	0.45925
	F_1-score	0.89062	0.66667	0.82625	0.65705	0.58225
R_2	Precision	0.92934	0.67607	0.82901	0.81931	0.60185
	Recall	0.74672	0.80126	0.69868	0.55720	0.57259
	F_1-score	0.82808	0.73336	0.75829	0.66329	0.58686

restriction, the summary could not repeat word grouping, resulting in more comprehensive summaries. The centrality values of the collected nodes were determined in the last stage using node centrality computations. This study employs eigenvector centrality to weigh the nodes of the textual graph (networks).

The experimental study is examined on BBC news article dataset and testing method is undertaken independently. It has been executed over 100-iterations for summaries of 100, 150, 200, 250, 300, and 350 words to completely analyze the proposed summarizing system's performance. The ROUGE performance measures are used to assess the proposed document summarization, as shown in Tables 4, 5, 6, 7, 8 and 9.

In 100 words length text summary, TGETS has the highest recall values than other existing state-of-the-art methods (i.e., TR, KH, LR, and PR) for ROUGE (R_1), as shown in Table 4. Similarly, TGETS has the highest precision values than other methods for ROUGE (R_2). However, F_1-score values corresponding to the ROUGE (R_1 and R_2) are 0.89062 and 0.82808, respectively.

In Table 5, all parameters (precision, recall, and F_1-score) corresponding to ROUGE (R_1 and R_2) are better for proposed approach (TGETS) than other existing state-of-the-art methods (i.e., TR, KH, LR, and PR).

In Table 6, the proposed approach produces better recall and F_1-score than existing state-of-the-art methods, but precision is not better than LR methods for both ROUGE (R_1 and R_2) metrics.

In Table 7, the proposed approach produces better results for evaluating ROUGE (R_1) for all parameters (Precision, recall, and F_1-score). Still, for the evaluation of R_2: precision values are lower than LR, and other methods, such as TR, KH, and PR, are better. Similarly,

Table 5 150-word length text summary (BBC News Article)

ROUGE Metric		TGETS	TR	KH	LR	PR
R_1	Precision	0.98013	0.71507	0.92414	0.84190	0.67321
	Recall	0.92500	0.60551	0.83750	0.65271	0.59203
	F_1-score	0.95177	0.65575	0.87869	0.73533	0.63002
R_2	Precision	0.94619	0.68105	0.86916	0.57334	0.51702
	Recall	0.89029	0.73525	0.78481	0.82163	0.57839
	F_1-score	0.91739	0.70711	0.82483	0.67539	0.54599

Table 6 200-word length text summary (BBC News Article)

ROUGE Metric		TGETS	TR	KH	LR	PR
R_1	Precision	0.73134	0.57021	0.71144	0.73628	0.69003
	Recall	0.91875	0.84950	0.89375	0.70032	0.57660
	F_1-score	0.81440	0.68238	0.79224	0.71785	0.62824
R_2	Precision	0.67320	0.69972	0.66667	0.80127	0.61433
	Recall	0.86919	0.72619	0.85232	0.69842	0.67884
	F_1-score	0.75875	0.71271	0.74815	0.74632	0.64498

Table 7 250-word length text summary (BBC News Article)

ROUGE Metric		TGETS	TR	KH	LR	PR
R_1	Precision	0.62331	0.49375	0.53744	0.59372	0.45000
	Recall	0.99285	0.74528	0.87142	0.76750	0.80000
	F_1-score	0.76584	0.59398	0.66485	0.66952	0.57599
R_2	Precision	0.58445	0.48484	0.48186	0.62626	0.45625
	Recall	0.95196	0.96000	0.81222	0.52100	0.89024
	F_1-score	0.72425	0.64429	0.60487	0.56880	0.60330

Table 8 300-word length text summary (BBC News Article)

ROUGE Metric		TGETS	TR	KH	LR	PR
R_1	Precision	0.60392	0.56102	0.57563	0.65532	0.53602
	Recall	0.96250	0.79113	0.85625	0.69023	0.77391
	F_1-score	0.74217	0.65649	0.68844	0.67232	0.63336
R_2	Precision	0.54293	0.47193	0.50409	0.54921	0.54791
	Recall	0.90717	0.72261	0.78059	0.69170	0.83785
	F_1-score	0.67931	0.57097	0.61258	0.61227	0.66255

Table 9 350-word length text summary (BBC News Article)

ROUGE Metric		TGETS	TR	KH	LR	PR
R_1	Precision	0.65258	0.53712	0.55932	0.60175	0.48570
	Recall	0.99285	0.88210	0.94285	0.77389	0.81073
	F_1-score	0.78753	0.66768	0.70212	0.67705	0.60747
R_2	Precision	0.60387	0.59188	0.52405	0.53007	0.49662
	Recall	0.95196	0.84721	0.90393	0.79389	0.85005
	F_1-score	0.73898	0.69689	0.66346	0.63569	0.62696

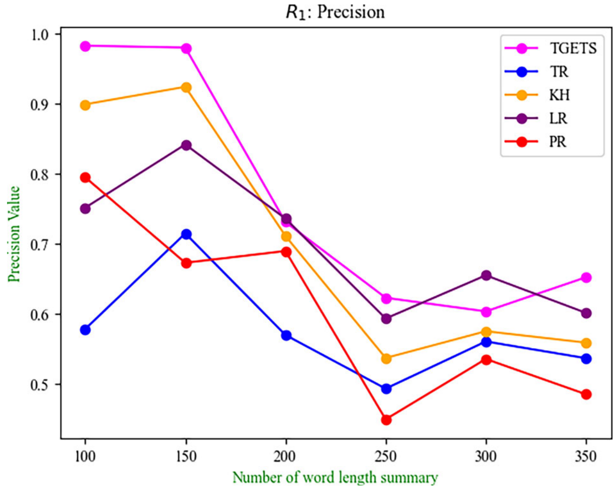


Fig. 7 Comparison of word length summary for parameter (Precision) under $ROUGE_1$

the recall value is less than TR, and better than others (LR, KH, and PR). Finally, the proposed approach produced good results as compared to other state-of-the-art methods.

For 300-word length text summary (Table 8), recall and F_1-score parameters corresponding to ROUGE evaluation for R_1 and R_2 are better than other state-of-the-art methods. Still, the method of LR is higher than the proposed method's precision values in ROUGE (R_1 and R_2) metric.

For 350-word length text summary (Table 9), the proposed approach produces better performance than other state-of-the-art methods (TR, KH, LR, and PR), in terms of precision, recall, and F_1-score for ROUGE (R_1 and R_2) metric.

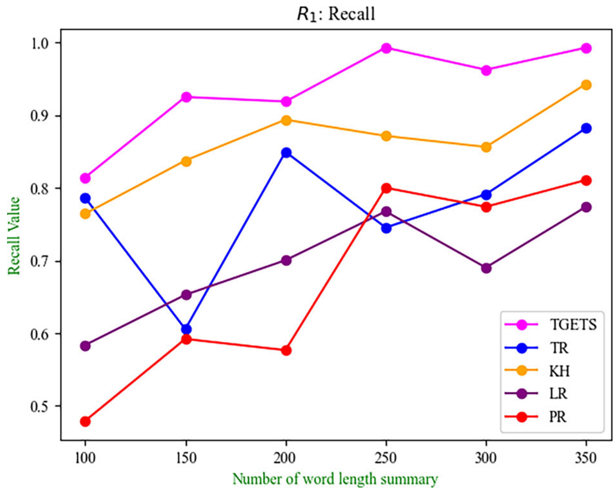


Fig. 8 Comparison of word length summary for parameter (Recall) under $ROUGE_1$

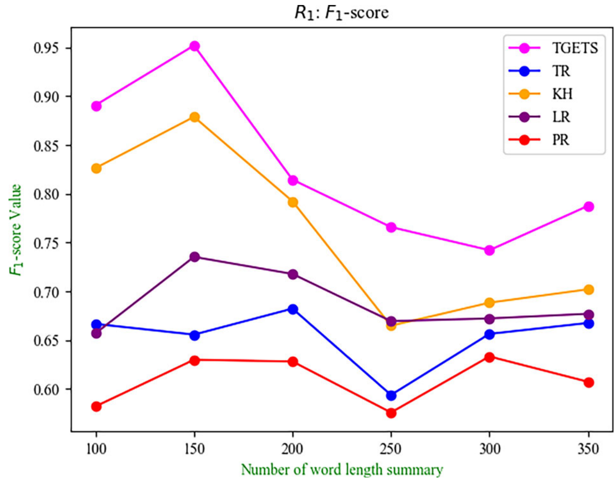


Fig. 9 Comparison of word length summary for parameter (F_1-score) under $ROUGE_1$

All tabular results are represented in graphical way format (in terms of precision, recall, and F_1-score) to compare all methods such as proposed (TGETS), TR, KH, LR, and PR.

The line chart plotting for the ROUGE (R_1) metric corresponds to the parameter of precision, recall, and F_1-score and the comparison between the proposed work and existing models, shown in Figs. 7, 8, and 9.

Figures 10, 11, and 12 represents the comparison of proposed and existing methods using the parameter of precision, recall, and F_1-score for the ROUGE (R_2) metric.

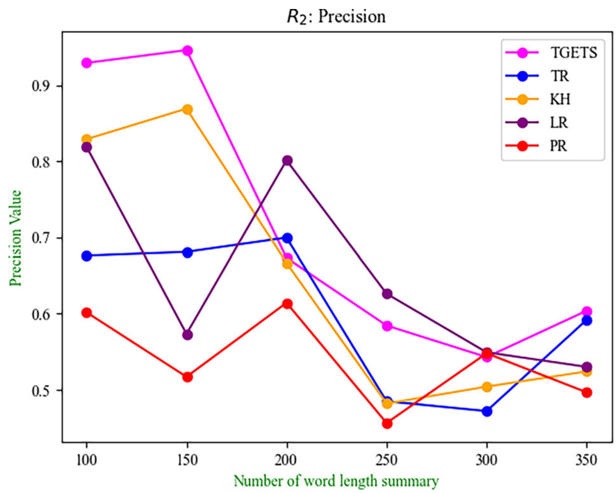


Fig. 10 Comparison of word length summary for parameter (Precision) under $ROUGE_2$

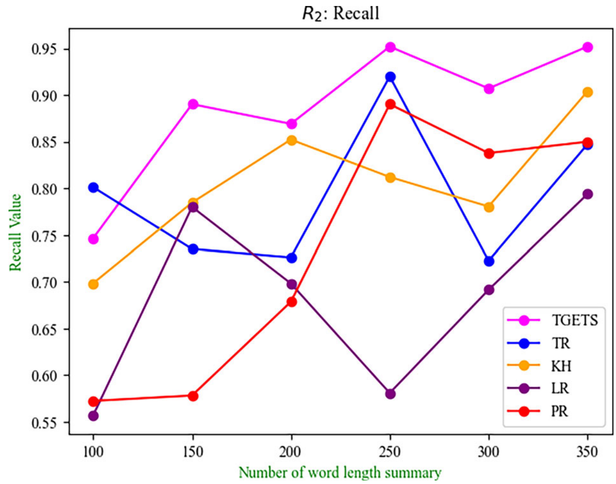


Fig. 11 Comparison of word length summary for parameter (Recall) under $ROUGE_2$

4.5 Comparison with existing research

We compare our experimental results with TR, KH, LR, and PR to evaluate the effectiveness on the extractive text summarization. From all summarizers' (generated summary (users) and provided a summary (given)), various ROUGE scores are shown in Tables 4–9 and have a value of ROUGE (R_1, R_2) corresponding to the parameter precision, recall, and F_1-score. The results in Tables 4 to 9 compare various summary techniques according to the caliber of the summaries they produce. First, we compare the ROUGE (R_1) metric for all size of word

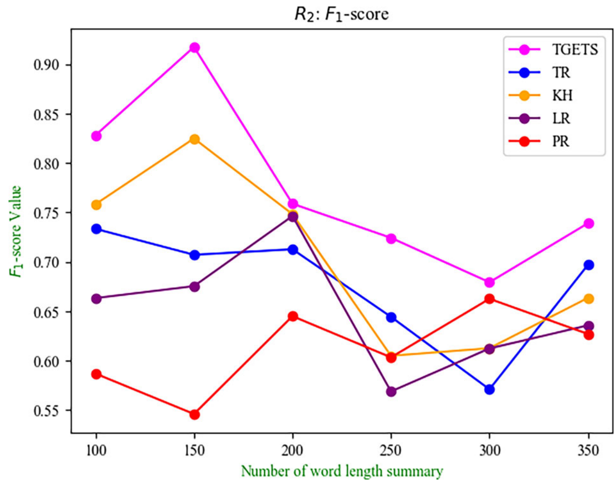


Fig. 12 Comparison of word length summary for parameter (F_1-score) under $ROUGE_2$

length summaries (100-, 150-, 200-, 250-, 300-, and 350-words) for the precision, recall, and F_1-score measurement in Figs. 7–9. In between 100 to 200-word length, the summary varies by 30.82% in the precision values which shows that the proposed method generate good results compared to the TR method. The summary's overall percentage variation under recall value is 38.76% than PR method and that the overall variation in the F_1-score for summaries of 100-200 words is 30.73% than PR method. After that, variation in between 250-350-word length text summary: precision varies by 15.13% than TR method, recall varies by 24.31% than LR method, and F_1-score varies by 20.86% than PR method. In ROUGE (R_1), overall summary length performance is improved than existing state-of-the-art methods.

For the ROUGE metric (R_2), 100-200 words length text summary; the precision, recall, and F_1-score measurement in Figs. 10–12 are 32% than PR methods, 27% than PR method, and 29% than PR method, respectively. Similarly to evaluate the variation for 250-350-words length text summary, precision, recall, and F_1-score exceeds by 13.35% than PR method, 28.62% than LR method, and 15.21% than LR method, respectively. The proposed method improved the performance of the summary than the existing state-of-the-art methods.

5 Conclusion and future directions

Nowadays, a large volumes of data continue to grow, and some textual features must be gathered by analyzing document data to reduce the time required to access information. This fascinating endeavor has piqued scholarly interest in automatic summarizing systems that can assist end users in this field. This paper proposes a new unsupervised ETS algorithm to achieve the goal. Using a lemmatization that applies text summarizing through a unique method avoids abnormalities in text documents in the ETS approach. Lemmatization yields positive results in generating discrete and quantitative linkages between sentences. Common terms represent the second finite set, whereas sentences reflect the first finite set in the representation of a graph. As a result, non-directional and weighted graphs were used to implement the proposed text representation method. The sentence weight and average weight of graph are calculated in order to eliminate less important sentences from the text document. The summarizing method used in the offered techniques produces values sorted in increasing order, and the amount of text data they carried about the strings that make up textual graphs is quantified. The datasets collected from BBC News articles are extensively used in this field to test the proposed methodology. The influence of exponential transformation on recall-oriented ROUGE-metric (R_1, R_2) performance measures is demonstrated. For summaries of 100, 150, 200, 250, 300, and 350 words, the effect of using a polynomial time complexity algorithm is investigated. On an average 100 to 200-words length text summary, the proposed approach (TGETS) $ROUGE_1$ value is better than other existing methods under the precision, recall and F_1-score, respectively. However, for $ROUGE_2$, 100 to 200-words length text summary perform better than others word length text summary in term of precision, recall and F_1-score. In the future, we will try abstractive text summarization using the LSTM techniques, neural network (NN), and fuzzy-logic-based approaches.

Appendix A: Text document and reference summary of BBC news articles

Sentences	Sentences of the text document
S_1	Young book fans have voted Fergus Crane, a story about a boy who is taken on an adventure by a flying horse, the winner of two Smarties Book Prizes.
S_2	Paul Stewart and Chris Riddell's book came top in the category for six to eight year olds and won the award chosen by after-school club members.
S_3	Sally Grindley's Spilled Water, about a Chinese girl sold as a servant, was top in vote of readers aged nine to 11.
S_4	Biscuit Bear by Mini Grey took the top award in the under-five category.
S_5	Winners were voted for by about 6,000 children from a shortlist picked by an adult panel.
S_6	The prize, which is celebrating its 20^{th} year, is billed as the UK's biggest children's book award.
S_7	Fergus Crane includes text by Stewart and illustrations by Riddell, who also created The Edge Chronicles together.
S_8	As well as the six to eight prizes, it won the 4-Children Special Award voted for by after-school club members.
S_9	Julia Eccleshare, chair of the adult judging panel, said children's literature had "never looked stronger" in the prize's 20 years.
S_{10}	This award counts because the final choice of winners is made by children, who are the toughest critics of all," she said.
S_{11}	This year's young judges chose the winners from an exceptionally strong and varied shortlist which showcases the very best in children's books today.
S_{12}	Previous winners have included JK Rowling, Jacqueline Wilson and Dick King-Smith.
Sentences	**Reference summary of text document**
S_1	Young book fans have voted Fergus Crane, a story about a boy who is taken on an adventure by a flying horse, the winner of two Smarties Book Prizes.
S_2	Paul Stewart and Chris Riddell's book came top in the category for six to eight year olds and won the award chosen by after-school club members.
S_6	The prize, which is celebrating its 20^{th} year, is billed as the UK's biggest children's book award.
S_8	As well as the six to eight prizes, it won the 4-Children Special Award voted for by after-school club member.
S_{11}	This year's young judges chose the winners from an exceptionally strong and varied shortlist which showcases the very best in children's books today.

Appendix B: Sample of system generated summary for BBC-news articles dataset from Appendix A text document

Sentences	System generated summary
S_1	Young book fans have voted Fergus Crane, a story about a boy who is taken on an adventure by a flying horse, the winner of two Smarties Book Prizes.
S_2	Paul Stewart and Chris Riddell's book came top in the category for six- to eight-year-olds and won the award chosen by after-school club members.
S_5	Winners were voted for by about 6,000 children from a shortlist picked by an adult panel.
S_7	Fergus Crane includes text by Stewart and illustrations by Riddell, who also created The Edge Chronicles together.
S_{11}	This year's young judges chose the winners from an exceptionally strong and varied shortlist which showcases the very best in children's books today.

Data availability The dataset is available on demand.

Declarations

Conflict of interest There is no potential for a conflict of interest.

References

1. Alami N, Mallahi ME, Amakdouf H, Qjidaa H (2021) Hybrid method for text summarization based on statistical and semantic treatment. Multimed Tools Appl 80:19567–19600
2. Al-Sabahi K, Zuping Z, Nadher M (2018) A hierarchical structured self-attentive model for extractive document summarization (HSAAS). IEEE Access 6:24205–24212
3. Awan MN, Beg MO (2021) Top-rank: a topical position rank for extraction and classification of key phrases in text. Comput Speech Lang 65:101116
4. Azadani MN, Ghadiri N, Davoodijam E (2018) Graph-based biomedical text summarization: an itemset mining and sentence clustering approach. J Biomed Inform 84:42–58
5. Bahloul B, Aliane H, Benmohammed M (2020) Ara* summarizer: an Arabic text summarization system based on subtopic segmentation and using an a* algorithm for reduction. Expert Syst 37(2):e12476
6. Belwal RC, Rai S, Gupta A (2021) A new graph-based extractive text summarization using keywords or topic modeling. J Ambient Intell Humaniz Comput 12(10):8975–8990
7. Brin S, Page L (1998) The anatomy of a large-scale hypertextual web search engine. Comput Netw ISDN Syst 30(1–7):107–117
8. Canhasi E, Kononenko I (2014) Weighted archetypal analysis of the multi-element graph for query-focused multi-document summarization. Expert Syst Appl 41(2):535–543
9. Chen Y-N, Huang Y, Yeh C-F, Lee L-S (2011) Spoken lecture summarization by random walk over a graph constructed with automatically extracted key terms. In: Twelfth Annual Conference of the International Speech Communication Association
10. Demange M, Di Fonso A, Di Stefano G, Vittorini P (2022) A graph theoretical approach to the firebreak locating problem. Theoret Comput Sci 914:47–72
11. Elbarougy R, Behery G, El Khatib A (2020) Extractive Arabic text summarization using modified pagerank algorithm. Egypt Inform J 21(2):73–81
12. El-Kassas WS, Salama CR, Rafea AA, Mohamed HK (2020) Edgesumm: graph-based framework for automatic text summarization. Inf Process Manag 57(6):102264
13. Erkan G, Radev DR (2004) Lexrank: graph-based lexical centrality as salience in text summarization. J Artif Intell Res 22:457–479

14. Fang C, Mu D, Deng Z, Wu Z (2017) Word-sentence co-ranking for automatic extractive text summarization. Expert Syst Appl 72:189–195
15. Fattah MA, Ren F (2009) Ga, mr, ffnn, pnn and gmm based models for automatic text summarization. Comput Speech Lang 23(1):126–144
16. Gambhir M, Gupta V (2017) Recent automatic text summarization techniques: a survey. Artif Intell Rev 47:1–66
17. Gambhir M, Gupta V (2022) Deep learning-based extractive text summarization with word-level attention mechanism. Multimed Tools Appl 81(15):20829–20852
18. Gupta VK, Siddiqui TJ (2012) Multi-document summarization using sentence clustering. In: 2012 4th International Conference on Intelligent Human Computer Interaction (IHCI). IEEE, pp 1–5
19. Hachey B, Murray G, Reitter D (2005) The Embra system at DUC 2005: query-oriented multi-document summarization with a very large latent semantic space. In: Proceedings of the Document Understanding Conference (DUC) 2005, Vancouver, BC, Canada
20. Jaradat YA, Al-Taani AT (2016) Hybrid-based Arabic single-document text summarization approach using genatic algorithm. In: 2016 7th International Conference on Information and Communication Systems (ICICS). IEEE, pp 85–91
21. Knight K, Marcu D (2002) Summarization beyond sentence extraction: a probabilistic approach to sentence compression. Artif Intell 139(1):91–107
22. Knor M, Škrekovski R, Yero IG (2022) A note on the metric and edge metric dimensions of 2-connected graphs. Discret Appl Math 319:454–460
23. Korenius T, Laurikkala J, Järvelin K, Juhola M (2004) Stemming and lemmatization in the clustering of finnish text documents. In: Proceedings of the thirteenth ACM international Conference on Information and Knowledge Management. pp 625–633
24. Krahmer E, Marsi E, van Pelt P (2008) Query-based sentence fusion is better defined and leads to more preferred results than generic sentence fusion. In: Proceedings of ACL-08: HLT, Short Papers. pp 193–196
25. Lloret E, Palomar M (2012) Text summarisation in progress: a literature review. Artif Intell Rev 37:1–41
26. Mahalleh ER, Gharehchopogh FS (2022) An automatic text summarization based on valuable sentences selection. Int J Inf Technol 14(6):2963–2969
27. Mallick C, Das AK, Dutta M, Das AK, Sarkar A (2019) Graph-based text summarization using modified textrank. In: Soft Computing in Data Analytics: Proceedings of International Conference on SCDA 2018. Springer, pp 137–146
28. Medelyan O (2007) Computing lexical chains with graph clustering. In: Proceedings of the ACL 2007 Student Research Workshop. pp 85–90
29. Miao L, Cao D, Li J, Guan W (2020) Multi-modal product title compression. Inf Process Manag 57(1):102123
30. Mihalcea R, Tarau P (2004) Textrank: bringing order into text. In: Proceedings of the 2004 Conference on Empirical Methods in Natural Language Processing. pp 404–411
31. Mihalcea R, Tarau P (2005) A language independent algorithm for single and multiple document summarization. In: Companion Volume to the Proceedings of Conference including Posters/Demos and tutorial abstracts
32. Moawad IF, Aref M (2012) Semantic graph reduction approach for abstractive text summarization. In: 2012 Seventh International Conference on Computer Engineering & Systems (ICCES). IEEE, pp 132–138
33. Moratanch N, Chitrakala S (2023) Anaphora resolved abstractive text summarization (AR-ATS) system. Multimed Tools Appl 82(3):4569–4597
34. Mutlu B, Sezer EA, Akcayol MA (2020) Candidate sentence selection for extractive text summarization. Inf Process Manag 57(6):102359
35. Nallapati R, Zhou B, Gulcehre C, Xiang B et al (2016) Abstractive text summarization using sequence-to-sequence RNNs and beyond. Preprint at http://arxiv.org/abs/1602.06023
36. Nandhini K, Balasundaram SR (2013) Improving readability through extractive summarization for learners with reading difficulties. Egypt Inform J 14(3):195–204
37. Nasar Z, Jaffry SW, Malik MK (2019) Textual keyword extraction and summarization: state-of-the-art. Information Processing & Management 56(6):102088
38. Parveen D, Ramsl H-M, Strube M (2015) Topical coherence for graph-based extractive summarization. In: Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing. pp 1949–1954
39. Plisson J, Lavrac N, Mladenic D et al (2004) A rule based approach to word lemmatization. In: Proceedings of IS, vol 3. pp 83–86
40. Rani R, Lobiyal D (2021) An extractive text summarization approach using tagged-LDA based topic modeling. Multimed Tools Appl 80:3275–3305

41. Roul RK (2021) Topic modeling combined with classification technique for extractive multi-document text summarization. Soft Comput 25:1113–1127
42. Sahoo D, Balabantaray R, Phukon M, Saikia S (2016) Aspect based multi-document summarization. In: 2016 International Conference on Computing, Communication and Automation (ICCCA). IEEE, pp 873–877
43. Salton G, Singhal A, Mitra M, Buckley C (1997) Automatic text structuring and summarization. Inf Process Manag 33(2):193–207
44. Song S, Huang H, Ruan T (2019) Abstractive text summarization using LSTM-CNN based deep learning. Multimed Tools Appl 78:857–875
45. Srivastava AK, Pandey D, Agarwal A (2021) Extractive multi-document text summarization using dolphin swarm optimization approach. Multimed Tools Appl 80:11273–11290
46. Srivastava R, Singh P, Rana K, Kumar V (2022) A topic modeled unsupervised approach to single document extractive text summarization. Knowl-Based Syst 246:108636
47. Steinberger J, Ježek K (2009) Update summarization based on latent semantic analysis. In: Text, Speech and Dialogue: 12th International Conference, TSD 2009, Pilsen, Czech Republic, September 13-17, 2009. Proceedings 12, Springer, pp 77–84
48. Tomer M, Kumar M (2022) Multi-document extractive text summarization based on firefly algorithm. J King Saud Univ - Comput Inf 34(8):6057–6065
49. Uçkan T, Karcı A (2020) Extractive multi-document text summarization based on graph independent sets. Egypt Inform J 21(3):145–157
50. Vaissnave V, Deepalakshmi P (2023) Modeling of automated glowworm swarm optimization based deep learning model for legal text summarization. Multimed Tools Appl 82(11):17175–17194
51. Wang D, Liu P, Zheng Y, Qiu X, Huang X (2020) Heterogeneous graph neural networks for extractive document summarization. Preprint at http://arxiv.org/abs/2004.12393
52. Xia T, Chen X (2020) A discrete hidden Markov model for SMS spam detection. Appl Sci 10(14):5011
53. Yadav AK, Maurya AK, Ranvijay, Yadav RS (2021) Extractive text summarization using recent approaches: A survey. Ingénierie des Systèmes d'Information 26(1)
54. Yadav AK, Ranvijay, Yadav RS, Maurya AK (2023) State-of-the-art approach to extractive text summarization: a comprehensive review. Multimed Tools Appl 1–63
55. Yadav AK, Saxena S (2016) A new conception of information requisition in web of things. Indian J Sci Technol 9:44