https://doi.org/10.48048/tis.2022.1719

Similitude Based Segment Graph Construction and Segment Ranking for Automatic Summarization of Text Document

Saravanan Arumugam[1,*] and Sathya Bama Subramani[2]

[1]*Department of Computing, Coimbatore Institute of Technology, Tamil Nadu 274001, India*
[2]*Independent Researcher, Tamil Nadu 274001, India*

([*]Corresponding author's e-mail: a.saravanan21@gmail.com)

Received: 9 August 2020, Revised: 29 April 2021, Accepted: 9 May 2021

Abstract

With the increase in the amount of data and documents on the web, text summarization has become one of the significant fields which cannot be avoided in today's digital era. Automatic text summarization provides a quick summary to the user based on the information presented in the text documents. This paper presents the automated single document summarization by constructing similitude graphs from the extracted text segments. On extracting the text segments, the feature values are computed for all the segments by comparing them with the title and the entire document and by computing segment significance using the information gain ratio. Based on the computed features, the similarity between the segments is evaluated to construct the graph in which the vertices are the segments and the edges specify the similarity between them. The segments are ranked for including them in the extractive summary by computing the graph score and the sentence segment score. The experimental analysis has been performed using ROUGE metrics and the results are analyzed for the proposed model. The proposed model has been compared with the various existing models using 4 different datasets in which the proposed model acquired top 2 positions with the average rank computed on various metrics such as precision, recall, F-score.

Keywords: Single document summarization, Graph-based summarization, Graph scoring, Sentence segment scoring, Information gain ratio

Introduction

Document summarization or text summarization is an important challenge to be addressed in the fields of information retrieval and natural language processing which are having various applications including speech recognition, machine translation, and video captioning [1,2]. Summarization is the process of representing the long text into a condensed format in such a way that the core meaning of the text is retained as original [3]. This process can be approached in 2 ways namely extractive and abstractive methods. Extractive summarization aims at selecting the significant segments or sentences from the document that express a core meaning of the document whereas, an abstractive summarization aims at paraphrasing the information in a concise manner that is not in the original text.

Generally, several algorithms were devoted to the literature for extractive summarization which extracts sentences based on their significance. However, many of the algorithms proposed are based on supervised learning approaches [4]. The first keyword extraction using a supervised learning approach was suggested by Turney [5,6]. The main drawback of the supervised learning approach in text summarization is that they are not cable of adapting new domains and required proper training for a specific domain.

The various models employed in document summarization are graph-based approaches [4,7], optimization-based approaches [8], integer programming [9] and neural network based approaches [10]. Variations in the neural networks with auto-encoder [11], recurrent neural network [12] and attentional encoder-decoder [13] are to name a few in neural networks based document summarization from the literature. Though several algorithms exist in generating abstractive summarization, extractive summarization is more attractive as the method is less complex, easy to implement, less time consuming, and produces better summaries with proper semantics.

In this paper, a general framework has been suggested to extract the summaries from the document which does not require any prior training. The model has 4 important components including data pre-processing, feature extraction, graph construction, and the segment scoring. Pre-processing is important for any text mining applications and it converts the given document into a particular format that is suitable for further processing in summary extraction. The role of feature extraction is to compute the scores for the segments for graph construction. It uses 3 main scores such as proximity based similarity score, weight based similarity score, segment significance score using information gain ratio, which are computed by comparing it with the document, title and terms in the segment, respectively. Graph construction is made by identifying the relationship between the segments using Minkowski distance computation between the extracted segment features. Finally, the scores are computed for all the segments based on which selection of segments for summaries are made by ranking them.

The experimental analysis has been made for the proposed model using 4 datasets namely CNN/Daily Mail Corpus, Daily Mail Corpus, DUC 2004 and Opinosis dataset. The proposed model is evaluated and compared with the existing models using the variants of ROUGE metrics. The results show that the proposed model outperforms than most of the existing complex models under comparison.

Related works

Automatic text summarization has become the most essential concept in today's digital era, as it allows the user to understand the documents easily and rapidly without reading the entire text. Several methods have been proposed by various authors in the field of document summarization. Litvak and Last [6] proposed a supervised as well as unsupervised approaches for extracting summary from the documents using graph-based text summarization. However, the method uses a directed graph. The combination of content-based and graph-based approaches with the Hopfield Network algorithm was introduced by Sornil and Gree-Ut [14]. Similarly, a graph reduction approach that utilizes the triangle counting was proposed by AL-Khassawneh *et al.* [15]. Unfortunately, these methods are not evaluated with the standard datasets. A new term weighting method was suggested by Mori [16] which can be utilized for computing the significance of the sentence in a cluster of documents in order to select them for the summary.

The neural network based approach has gained attention for improving summarization results. Rush *et al.* [3], suggested a local attention-based model for abstractive summarization and evaluated the results through DUC 2004 dataset. The work was improvised by replacing the decoder with Recurrent Neural Network (RNN) for better output [17]. Cheng and Lapata [13] introduced an attentional encoder-decoder mechanism for extractive summary and trained the model using the Daily Mail Corpus. Following the work, Nallapati *et al*. [18] introduced classifier and selector based RNN for an extractive summary. The authors also presented the work for abstractive summarization using attentional encoder-decoder with RNN and evaluated the results using DUC Corpus and CNN/Daily Mail Corpus [1,12]. As an extension, a simple recurrent network-based sequence classifier termed as SummaRuNNer was introduced and is evaluated through CNN/Daily Mail Corpus [18].

A novel architecture was introduced using a standard sequence-to-sequence attentional model with a hybrid pointer-generator network and coverage based mechanism by See *et al*. [19]; Tan *et al*. [20] attempted the combination of an extractive and abstractive summary using an attention-based decoder. Chen *et al*. [21] proposed a unified framework for an abstractive summary from the extractive summary using multi-tasking, and the results were evaluated using CNN/Daily Mail dataset. However, in general, the methods take longer execution time due to its complex architecture. Several linguistically inspired sentence compressions [22] and makeover of input sentences [9] for abstractive summarization were also modeled in previous works. Phrase-based statistical translation schemes for machines to produce summaries were also proposed and were devoted to the literature [23,24].

Kågebäck *et al*. [11] introduced a sentence representation using continuous vectors with the base of similarity measurement. The method was evaluated using the Opinosis dataset created and donated by Ganesan *et al.* [25]. Lin and Bilmes [26] suggested the summarization method based on optimization techniques using submodular functions. Several word embedding was introduced by several authors [27-29]. Similarly, an unfolding recursive auto-encoder (RAE), for phrase embedding was suggested by Socher *et al.* [10] to encode compressed sentences. By modifying the combinations of aforementioned word embedding, phrase embedding, and baselines, Kågebäck *et al.* [11] analyzed the models providing good results. However, the results are analyzed with the small dataset. Several other related works are also available in the literature. On analyzing the existing models, it is understood that the models are very complex in architecture, thereby, increasing the complexity of implementation and execution time. To

overcome the drawbacks of the existing models, the proposed model has been suggested with simplicity as the main concern.

Materials and methods

The idea of the proposed similitude document graph based text summarization model is inspired by the graph based summarization method [14,15]. **Figure 1** shows the overall steps in the proposed similitude graph based document summarization model.

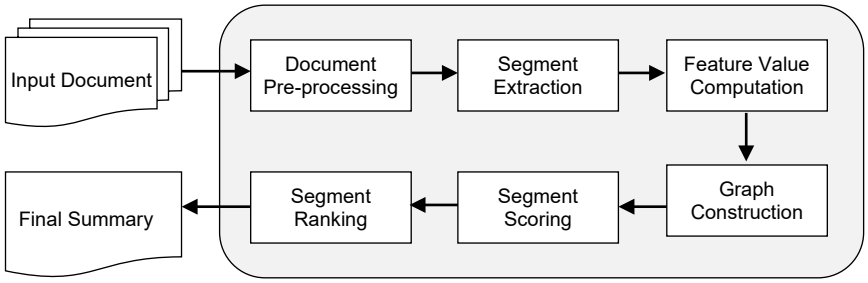

Figure 1 Overall architecture of the proposed model.

Though it resembles the general idea of the existing graph based summarization model, the proposed model introduces new procedures at various steps including feature extraction, graph construction, and segment scoring. The detailed architecture of the proposed similitude graph based document summarization model is presented in **Figure 2**. The details of each component are explained in detail in the below sub-sections.

Phase 1: Document pre-processing

The documents are pre-processed initially to represent the text in the document as a specific form that is suitable for analyzing them using the proposed similitude graph based model [30]. This pre-processing phase is essential for any text mining applications. Once the text in the document is transformed into a specific form, then several algorithms can be applied over it to analyze the text. The following are the pre-processing steps carried out in the proposed work.

Title identification

When compared with the normal text present in the document, the title text is considered to be the most significant and most informative [30]. Thus, the title and subtitles present in the document are identified and are marked with a specific identification to separate them from other text. To identify the titles and subtitles in the text, the following heuristics are employed.

1) The words in the capital letters are identified as titles and the few set of words with first letter capital is considered as the subtitle.

2) The segment of text without any ending symbols such as ',' (comma) or '.' (full stop) are considered to be the subtitles.

3) Sometimes the subtitles can be identified if the segment ended with a symbol ':' (colon).

Paragraph identification

The next step is the paragraph extraction in which the entire document is partitioned based on the paragraph. This step is essential for evaluating the significance of the sentence segments. The simple heuristics used here is that the set of sentences that are separated from other text using carriage return can be identified as a paragraph. Thus, each paragraph is identified using an ID $< P_0, P_1, P_2, \ldots P_n >$ where n is the total number of paragraphs in the text document.

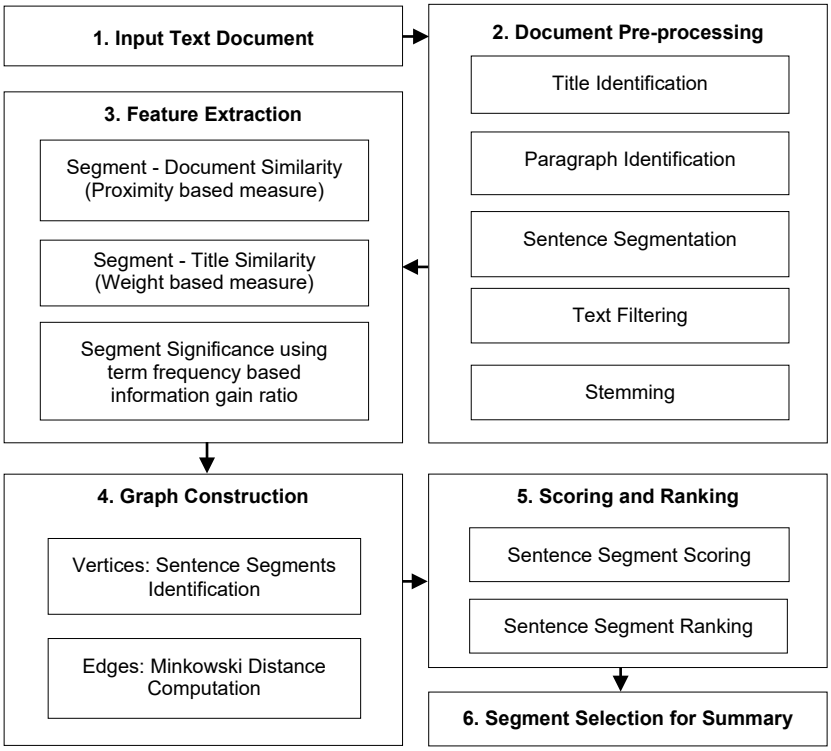

Figure 2 Detailed architecture of the proposed similitude graph based summarization model.

Sentence segmentation

Once the paragraphs are identified, the next step is to split the sentences in each paragraph as segments and is termed as sentence segments. Several criteria exist in splitting the segments with variable length. Usually the punctuation marks such as "." (full stop), "?" (question marks), "," (comma), "!" (exclamation marks) and carriage return can be taken as a delimiter for segmenting the sentences. Care must be taken in segmenting the sentences using the delimiter "." (full stop), since it can be identified with diverse meanings such as in email id or with a surname which must be removed or modified before segmenting the sentences.

Text filtering

Text filtering is applied to eliminate the insignificant data present in the document. The given input document may contain various types of data other than text like images and they must be removed initially. Also the text data may contain several other stop words such as articles, preposition, connectors, interjection, auxiliary noun, and auxiliary verbs. These words occur again and again in the document but have a less significant meaning. Thus, the stop words are to be removed for obtaining more accurate results [31,32]. Other than stop words, the text may include some other data such as numbers, symbols (@, #, $, %, *), and text inside the brackets that specifies additional meaning, which must also be removed before processing further. Several stop word lists are available on the web, based on which the text can be filtered.

Stemming

The last step in pre-processing is the reduction of terms. This can be done by applying stemming which shrinks the word in the segments to its root form [33]. For example, words such as filtering, filters, filtered, are all reduced to its root form as filter. This method reduces the variants of the word conveying the same meaning. The proposed model makes use of the suffix removal algorithm proposed by porter and is popularly known as porter stemming algorithm [34].

Once the pre-processing phase is executed, the proposed model extracts the features based on several similarity measures. The similitude graph is constructed with segments as vertices and the similarity between the segment features as edges that are computed using Minkowski distance. Finally,

the segments are scored and ranked with which the top-ranked segments are selected for the text summary.

Phase 2: Feature extraction

On applying the statistical and mathematical analysis of the segments extracted from the document, several substantial information can be collected. This information reflects the significance of the segments in the document. Feature extraction is the next important step which analyses the segments based on their position and occurrences concerning the other segments in the documents. The proposed model makes use of 3 feature extraction measures such as a) proximity based similarity between segment and document b) weight based similarity between segment and title c) segment significance using information gain ratio. These 3 measures result in the numerical values which act as a numerical feature vector and are represented as <F1, F2, F3>. These measures are explained below.

Proximity based segment - document similarity

In this measure, the closeness of the terms is considered along with the frequency of the terms as it statistically estimates the appropriate significance of the terms in the segments. The main theory behind the proximity based measure is that the degree of closeness is directly proportional to the relevance score of the segment in the document [35]. The feature is extracted by pairwise proximity between the term pair present in each segment. Generally, the closeness of the term-pair and the corresponding frequencies influence the significance of the segments. The model presented in [35] is employed here with little modification.

For a set of terms in the segment, the position vectors are extracted. Consider the segment S_1 = {a, b, c}. The set of terms in the document D excluding the segment S as T_D = {a b c d a b a c a b d e}. The position for the terms in the document can be considered as P_D = {1 2 3 4 5 6 7 8 9 10 11 12}. Thus, the position vector for each term in the segment S can be represented as $P_S(a)$ = {1, 5, 7, 9}, $P_S(b)$ = {2, 6, 10}, $P_S(c)$ = {3, 8}. The frequency of the corresponding segment terms in the document is $f_S(a) = 4$, $f_S(b) = 3$ and $f_S(c) = 2$. With these information, the first numerical feature value, a proximity based segment relevance score can be computed as in Eq. (1);

$$F_1(S) = \frac{\sum_{i=1}^{k}(f(t_i) + \sum_{j=1,j\neq i}^{n} TP(t_i, t_j))}{N} \quad (1)$$

here, $f(t_i)$ represents the frequency of term t_i in the segment, k is the number of terms in the segment and $TP(t_i, t_j)$ represents term proximity of each term t_i with all the other terms t_j adjacent to each other in the segment. N represents the total number of occurrences of the segment terms in the document and n represents the number of term pairs for the term t_i. The term proximity can be computed as in Eqs. (2) - (3);

$$TP(t_i, t_j) = \frac{1}{\min\{dis(t_i, t_j)\}} \quad (2)$$

where,

$$dis(t_i, t_j) = P(t_j) - P(t_i) \quad (3)$$

The relevance measure is evaluated for all the segments present in the document. The algorithm for measuring the proximity based similarity between segments and the document is presented in **Figure 3**.

Input: Set of segments extracted from the document after pre-processing S_i
Output: Feature element of the segments
Begin
N = 0; F
For each S_i in the document
  For each term t_j in the segment S_i
    f[t_j] = 0, P[t_j] = {ϕ};
    For each term t_k in the other segments
      If $t_j = t_k$ **then**
        f[t_j] = f[t_j] + 1;
        p[t_j] = p[t_j] ∪ j;
        N = N + 1;
      End If
    End For
  End For
  For each term t_j in the segment S_i
    Total_TP = 0, PTF = 0;
    For each term t_l in the segment S_i and $l \neq j$
      Read other segments to compute the distance between the term pairs
        $TP(t_j, t_l) = \frac{1}{\min\{p(t_l) - p(t_j)\}}$;
        Total_TP = Tota1_TP + $TP(t_j, t_l)$;
    End For
    PTF = PTF + Total_TP + f[t_j];
  End For
  $F_1(S_i)$ = PTF / N;
End For
End Algorithm

Figure 3 Algorithm for proximity based segment-document similarity.

Weight based segment - title similarity

In the second feature extraction measure, the similarity between the segments and the title and subtitles of the document is evaluated in order to identify the relevance score of the segments. In this method the weight is assigned to the terms and its frequencies based on the occurrences of the segment terms in the titles and normal content and finally, the obtained values are normalized [36].

Each term in the segment is compared with all the title or subtitle terms and the count is increased for each occurrence and is termed as a positive hit count. Similarly, each term in the document is compared with the normal text of the document and if the match occurs, the count is increased for each occurrence and is termed as negative hit count. This is carried out for all the terms in the segment. Finally, the weight is applied for the positive hit count and negative hit count, and the values are normalized to obtain the relevance score of each term. Generally, the titles and subtitles have higher weights than other terms. The evaluation is done for all the segments in the document. The formula to compute the weight based similarity between the segment and the title is shown in (4);

$$F_2(S) = \frac{\sum_{i=1}^{n}\{\alpha * P_{hit}(t_i) + \beta * N_{hit}(t_i)\}}{\sum_{i=1}^{n}(P_{hit}(t_i) + N_{hit}(t_i))} where\ \alpha > \beta \quad (4)$$

here n represents the number of terms in the segment S. $P_{hit}(t_i)$ represents the number of occurrences of term t_i in the title of the document and $N_{hit}(t_i)$ represents the number of occurrences of term t_i in the normal text of the document. α and β denotes the weights with the constrain that $\alpha > \beta$. In the proposed model, the value of the variable α is assigned as 1 whereas the value of the variable β is assigned as 0.5.

The evaluation measure is computed for all the segments in the document which is the second element in the feature vector of the segments. The algorithm for measuring the weight based similarity between the segments and the title is presented in **Figure 4**.

Input: Set of segments extracted from the document after pre-processing S_i
Output: Feature element of the segments
Begin
$N = 0$, $P_{hit} = 0$ $N_{hit} = 0$;
For each S_i in the document
  $P_{hit}(t_i) = 0$, $N_{hit}(t_i) = 0$;
  For each term t_j in the segment S_i
    For each term t_k in the title/subtitle
      If $t_j = t_k$ **then**
        $P_{hit}(t_i) = P_{hit}(t_i) + 1$;
      End if
    End For
    For each term t_l in the other segment
      If $t_j = t_l$ **then**
        $N_{hit}(t_i) = N_{hit}(t_i) + 1$;
      End If
    End For
    $P_{hit} = P_{hit} + P_{hit}(t_i)$;
    $N_{hit} = N_{hit} + N_{hit}(t_i)$;
  End For
  $F_2(S_i) = (P_{hit} + (0.5 \times N_{hit})) / (P_{hit} + N_{hit})$;
End For
End Algorithm

Figure 4 Algorithm for weight based segment-title similarity.

Segment significance computation

The next feature extraction involves the computation of sentence significance. Based on the idea proposed by [16], the proposed model modifies the concept of information weight ratio based term weighting for the evaluation of sentence significance. The idea is to represent the document in a tree structure in such a way that the root node is the entire document, the next level node has the paragraphs followed by the segment level and finally, the leaf node constitutes the terms. The sample tree representation of the document is shown in **Figure 5**.

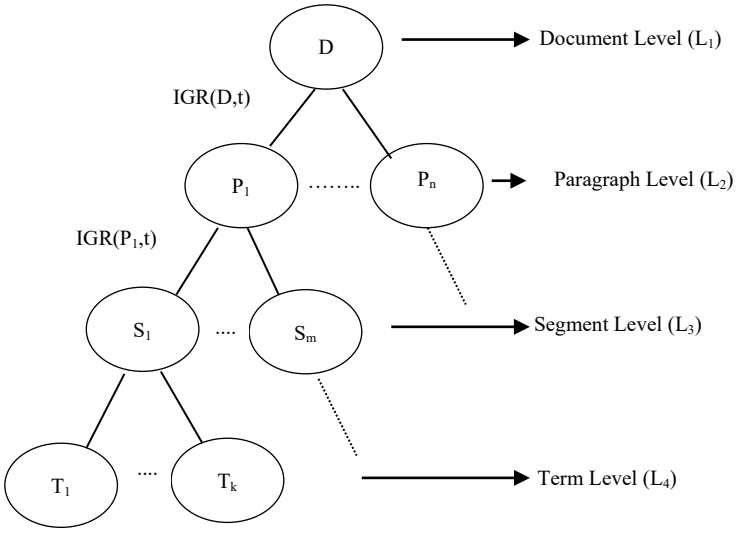

Figure 5 Tree representation of a document.

The information gain ratio is computed for each term in the segments based on which the weight is assigned to the terms and these weights are employed to compute the significance of the segments. The idea is to compute the information gain ratio of each term in the segment at each level from the root node and move down along with its sub-level to the corresponding term. The final weight of the term is the sum of all the information gain ratio obtained at each level. The details of the computation of information gain ratio of a term t at each level L represented as IGR(L, t) which is based on the idea given by Mori [16] is explained below;

$$IGR(L,t) = \frac{Gain(L,t)}{Split\_Info(L)} \tag{5}$$

$$Gain(L,t) = Entropy(L,t) - Entropy_{SL}(L,t) \tag{6}$$

here SL represents the sublevel partition of the particular node.

$$Entropy(L,t) = -\frac{f(t,L)}{|L|}\log_2\frac{f(t,L)}{|L|} - \left\{\left(1-\frac{f(t,L)}{|L|}\right)\log_2\left(1-\frac{f(t,L)}{|L|}\right)\right\} \tag{7}$$

where $f(t)$ represents the number of occurrences of the term t in the corresponding node in level L and $|L|$ represents the total number of terms in the level L;

$$Entropy_{SL}(L,t) = \sum_i \frac{|L_i|}{|L|} Entropy(L_i,t) \tag{8}$$

$$Split\_Info(L) = -\sum_i \frac{|L_i|}{|L|} Log_2\frac{|L_i|}{|L|} \tag{9}$$

here L_i represents the i^{th} sub-level partition of level L. Finally, the weight of the term t is computed by summing all the IGR values obtained at each level from the root;

$$Weight(t) = \sum IGR(L,t) \tag{10}$$

For example, in **Figure 5**, the weight or information gain ratio of t_1 is obtained from the root node D, IGR(D, t_1) for which the sub-levels to be considered are P_1, P_2, .., P_n. At the next level, the information gain ratio of node P_1 in which the term t_1 is present, IGR(P_1, t_1) is obtained from the sub-levels S_1, S_2, ..,

S_m. Thus finally the weight of the term t_1 is obtained by summing all the information gain obtained above from the root node;

$$Weight(t_1) = \text{IGR}(D, t_1) + \text{IGR}(P_1, t_1)$$

On computing, the weight of the terms presents in the segments, the significance of the segments can be obtained by summing the product of the term weights and its frequencies. The formula to compute the significance of a segment S is presented in Eq. (11);

$$F_3(S) = \frac{\sum_i weight(t_i) \times f(t_i, D)}{N} \quad (11)$$

Here, i represents the i^{th} term in the segment S, $f(t_i, D)$ is the frequency of the term t_1 in the document, and N represents the total number of terms in the segment S. The algorithm for computing the segment significance is shown in **Figure 6**. On extracting the features using methods presented in above subsections, the next step is evaluated by constructing the document graph for scoring the segments effectively.

Input: Set of segments extracted from the document after pre-processing S_j along with the paragraph details P_k, Total number of terms N in the document
Output: Feature element of the segments
Begin
Construct the document tree;
For all the segment in the document
  $F_3(S) = 0$;
  For each term t_i in the segment Sj
    Compute frequency of the term t_i in document D as $f(t_i)$;
    Traverse the tree from root to leaf node t_i;
    $W(t_i) = 0$;
      For each Level L in the tree
        Compute term frequency of term t_i and total terms as in (7);
        Compute the entropy of the corresponding sublevel;
        $E_{SL}(L, t_i) = \sum_i \frac{|L_i|}{|L|} E(L_i, t_i)$;
        Compute the information gain at the level L for the term t_i;
        $Gain(L, t) = E(L, t_i) - E_{SL}(L, t_i)$;
        Compute the split information for level L;
        $Split\_Info(L) = -\sum_i \frac{|L_i|}{|L|} Log_2 \frac{|L_i|}{|L|}$;
        Compute the information gain ratio Level L;
        $IGR(L, t_i) = \frac{Gain(L,t_i)}{Split\_Info(L)}$;
        $W(t_i) = W(t_i) + IGR(L,t_i)$;
      End For
    $F_3(S) = F_3(S) + (W(ti) \times f(t_i))$;
  End For
$F_3(S) = F_3(S)/N$;
End For
End Algorithm

Figure 6 Algorithm for computing the significance of the segment.

Phase 3: Graph construction

In this phase, the segments in the document are represented as an undirected graph data structure. In the feature extraction phase, 3 features such as proximity based similarity, weight based similarity, and the segment significance are extracted for all the segments in the document. Thus each segment S_i is represented as a feature vector denoted as $< F_{1i}, F_{2i}, F_{3i} >$. The segment graph representation of the document involves a set of vertices and edges. Each vertex represents the segments in the document and the edges represent the relationship between the segments which is evaluated based on the similarity between them.

Let G = < V, E > be the document graph where, V represents vertices representing the segments in the document {S1, S_2, …S_n} and E represent the edges that denotes the similarity between the feature vectors of the segments. Several methods exist in computing the similarity between the segments. In the proposed model, the edges representing the similarity between the segments are computed using the Minkowski distance measure.

The general formula to compute the Minkowski distance between 2 vectors X and Y of n dimensions is given in (12);

$$D(X,Y) = \left(\sum_{i=1}^{n} |x_i - y_i|^p\right)^{\frac{1}{p}} \tag{12}$$

This approach is a generalized metric as the distance between 2 vectors can be computed using different ways by varying the value of p, however, the triangular inequality property is satisfied only when $p \geq 1$. The parameter p that varies between $0 \leq p \leq 1$ is used widely in several applications. For instance, when the value of p is 1, the method becomes Euclidean distance; when the value of p is 2, the method becomes Manhattan distance. In some situations, the parameter p tends to ∞ which leads to Chebychev distance.

Though Euclidean distance and Manhattan distance are used most widely in several applications [37], in some situations, Euclidean distance underestimates the point vectors and the Manhattan distance metric overestimates the point vectors which may lead to inaccurate results [38].

Thus, to overcome the drawbacks of the Euclidean distance and Manhattan distance, the optimal value for the parameter p is chosen which is a combination of the 2 aforementioned metrics with roughly equal distributions for each of them. The value of the parameter p is chosen as an intermediate value (p = 1.5) that lies between Manhattan distance (p = 1) and Euclidean distance (p = 2).

Owing to the above idea, the formula to compute the similarity between the 2 segment feature vectors S_i and S_j specified as (F_{1i}, F_{2i}, F_{3i}) and (F_{1j}, F_{2j}, F_{3j}), respectively is given in (13);

$$D(S_i,S_j) = \left(|F_{1i} - F_{1j}|^{1.5} + |F_{2i} - F_{2j}|^{1.5} + |F_{3i} - F_{3j}|^{1.5}\right)^{1/1.5} \tag{13}$$

Based on the computed distance between all the segment feature vectors, the document graph is constructed to select the segments for the summary. The sample document graph DG with 5 segments S_1, S_2, S_3, S_4, and S_5 is presented in **Figure 7**.

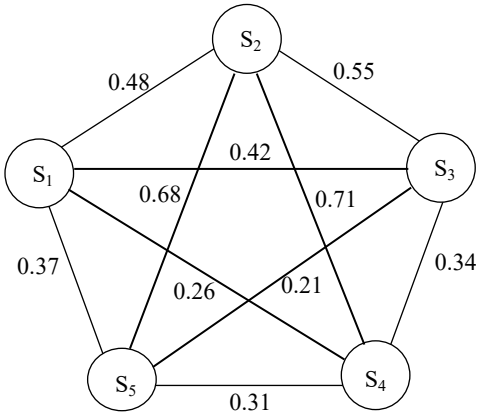

Figure 7 Sample segment graph of a document.

After constructing the document graph, the next step is to evaluate the scores for the segments with which the segments are selected for the summary. The details about the segment scoring mechanism are presented in the next section.

Phase 4: Segment scoring and ranking

Scoring and ranking are predominant functions in several applications [39]. In selecting the segments for the summary, scoring and ranking are the 2 main steps to be executed for effective results. This involves the edge values computed for the undirected document graph. Consider the document graph DG with n vertices and m edges. The overall score of the document graph is computed as in Eq. (14);

$$Score(DG) = \frac{\sum_{j=1}^{m} value(E_i)}{m} \tag{14}$$

here, $value(E_j)$ represents the edge values of the graph, and m is the number of edge values. To compute the score for each vertex (segment), the degree of each vertex is employed where degree represents the number of edges associated with the corresponding vertex. The formula to compute the score for the segment S_i is given in Eq. (15);

$$Score(S_i) = Score(DG) - \left(Cos\left(\frac{\sum_{j=1}^{k} value(E_i)}{k} \right) \times \frac{n-1}{m} \right) \tag{15}$$

here k represents the degree of the vertex S_i, n represents the number of vertices, m denotes the number of edges. Also, the values are normalized based on the number of nodes and edges. On computing the scores for the vertices, the vertices are sorted in descending order of their scores. Finally, the segments having top rank is selected for the document summary. The algorithm pseudocode for graph construction and segment scoring and selection are presented in **Figure 8**.

Input: Segment feature vectors for all the segments in the set S = {S1, S_2, …S_n}
Output: Segment selected for summary
Begin
Function Graph_Construction()
D[n][m] = 0 //Edge Values
For each segment S_i (F_{1i}, F_{2i}, F_{3i}) in segment set S
//S_i represents the vertex in the graph
For each segment S_j (F_{1j}, F_{2j}, F_{3j}) in S
If i < j **then**
Compute edge values of vertex S_i with vertex S_j;
End If
End For
End For
Segment_Scoring (S, D);
End Function
Function Segment_Scoring (S, D)
n = number of vertex, m = number of edges;
For each segment S_i in the segment set S
For each segment S_j in the segment set S
Compute the sum of edges;
End For
End For
$Score(DG) = Average\ of\ edge\ values$;
For each segment S_i in the segment set S
For each segment S_j in segment set S and i ≠j
Compute the sum of edges;
End For
Compute the segment score;
$Score(S_i) = Score(DG) - \left(Cos(avg(E)) \times \frac{n-1}{m}\right)$;
End For
Select the segments having highest scores;
End Function
End Algorithm

Figure 8 Algorithm steps for segment ranking using document graph.

To illustrate the computation of vertex or segment score, consider the document graph in **Figure 7**. The score for the document graph DG is;

$$Score(DG) = \frac{4.33}{10} \Longrightarrow 0.433$$

Now, the score for the vertex S_1 with degree 4 is computed as

$$Score(S_1) = 0.433 - \left(Cos\left(\frac{0.48 + 0.42 + 0.26 + 0.37}{4}\right) \times \frac{4}{10}\right)$$

$$Score(S_1) = 0.433 - \left(Cos(0.383) \times \frac{4}{10}\right)$$

$$Score(S_1) = 0.433 - \left(0.928 \times \frac{4}{10}\right) \Longrightarrow 0.433 - 0.371 \Longrightarrow 3.959$$

Similarly, the scores for other segments S_2, S_3, S_4, and S_5 are also computed and the values are 4.001, 3.959, 3.962 and 3.96, respectively. The order of segments after ranking based on the obtained scores are S_2, S_4, S_5, S_1 and S_3. Thus, the top-ranked segment S_2 is selected for the document summary. Additionally, the next ranked segments S_4 and S_5 can also be selected for the summary based on the requirement of the length of the summary.

Results and discussion

This section presents the extensive experimental analysis made for the proposed system along with the details about the dataset used, evaluation metrics, experimental results obtained along with the performance analysis.

Dataset used

To evaluate the performance of the proposed system, several datasets have been employed. The CNN/Daily Mail dataset has been utilized for summarization that comprising online news articles from CNN and Daily Mail news agencies, a non-anonymized version [1,19,40]. The dataset is made available by Kyunghyun Cho is an academic at New York University. There are about 93,000 documents termed as stories approximately extracted from CNN and 1,97,000 news articles from Daily Mail. On average the documents and the summaries consist of 28 sentences and 4 sentences, respectively [2].

Document Understanding Conferences (DUC) is another popularly utilized corpus for text summarization application in which the documents and summary pairs are created from the past workshop of DUC by the National Institute of Standards and Technology (NIST). This dataset can be easily used for single document summarization. For evaluation, DUC 2004 is used which consists of 500 news articles extracted from various news agency websites and is summarized at 75 bytes. However, the corpus can be used only for testing purposes and cannot be utilized for training as they are too small to train the underlying model [1]. The dataset is available at the NIST website (https://duc.nist.gov/).

Another freely available dataset used in this work is Opinosis dataset. The dataset is created by extracting the user reviews on a specific topic comprising of 50 topics with 100 sentences each on average. The data for the corpus is collected from various sources about different products specifically for text summarization [11,25]. The list of Corpora repositories used in the existing methods is presented by Dernoncourt *et al.* [41].

Based on the information in the datasets, the pre-processing of text is made. All the documents are analyzed for data cleaning. The tables and figures presented in documents are removed. The titles in the documents are identified. The paragraph and the sentences are also extracted. The text is filtered by removing stop words and finally stemming is applied over the data using the porter stemmer algorithm.

Performance metrics

There are several evaluation metrics available among which ROUGE metrics have been considered as the best among them. In the evaluation of the proposed model, ROUGE measures with various parameters are employed for analyzing the summaries.

ROUGE metric

To evaluate the performance of the proposed single document summarization model and to compare it with the existing models, the obtained results are analyzed using the ROUGE (Recall-Oriented Understudy for Gisting Evaluation) proposed by Lin [42]. The main idea is to compare the extracted summary with that of the reference summary that is ideally created by humans. The evaluation is based on the number of overlapping units which include n-grams (unigram, bigram), word sequence, and word pairs. Some of the ROUGE metrics used for the evaluation are ROUGE-N, ROUGE-L, and ROUGE-SU (skipped bigrams).

The Recall based ROUGE-N score between the generated summary and the reference summary is given as in Eq. (15);

$$R_{ROUGE-N} = \frac{\sum_{S \in RS} \sum_{gram_n \in S} Count_{match}(gram_n)}{\sum_{S \in RS} \sum_{gram_n \in S} Count(gram_n)} \quad (15)$$

here n specifies the length of n-grams denoted as $gram_n$. Generally, n takes the value as 1 for unigrams and 2 for bigrams. $Count_{match}(gram_n)$ denotes the number of n-gram matches between the generated summary and the reference summary. The denominator denotes the number of n-grams in the referenced

summary as it is a recall-related metric. In the case of precision based ROUGE metrics ($P_{ROUGE-N}$), the denominator can be modified as the number of n-grams occurred in the generated summary instead of considering a reference summary. Based on the precision and recall value, the f-score can be computed as in Eq. (16);

$$F_{ROUGE-N} = \frac{(1+\beta^2)R_{ROUGE-N}.P_{ROUGE-N}}{R_{ROUGE-N}+\beta^2 P_{ROUGE-N}} \quad (16)$$

In the case of ROUGE-L, the longest common subsequence, the ROUGE metrics is computed between the 2 sequences X and Y, by measuring the common subsequence of maximum length. For ROUGE-S (skip bigrams), the ROUGE scores are computed between the generated summary and the reference summary by skipping the bigrams in which skip bigrams are the pair of terms in the sentence order, that allows arbitrary gaps between the term pairs. The method has been extended as ROUGE-SU in which it uses unigram along with the skip bigrams. The skipped bigrams can also be varied in the number of intermediate terms between the term pair. For example, ROUGE-SU4 allows 4 intermediate terms between the term pairs.

Average rank

With the results obtained by the ROUGE metrics for the proposed method and existing method, it is difficult to analyze the performance of the model as several variants of the ROUGE-metrics are used with each dataset. Thus, for analyzing the performance easily, the average rank is computed for the models under comparison. Average ranking is one of the most popularly and effectively used metrics in several applications. It identifies the average rank of the model obtained for various ROUGE results. Based on the values obtained, the ranks are assigned accordingly for all the models with respect to each variant of ROUGE measures. The ranks and ROUGE scores are inversely proportional in such a way that the highest ROUGE score is assigned a rank as 1, second highest as 2, and so forth. Finally, for each model, the ranks obtained for each metric are averaged to identify the performance position among others. The evaluation results for the 4 datasets are presented in the next subsection.

Evaluation results

Results on Daily Mail Corpus

As the experimental analysis made by the authors Cheng and Lapata [13]; Nallapati *et al.* [2], the proposed method is evaluated at a limited length ROUGE recall at 75 bytes of summary and 275 bytes of summary on Daily Mail dataset. At each recall, the unigram (ROUGE-1), bigram (ROUGE-2) overlap for evaluating the degree of information, and the longest common sequence (ROUGE-L) for evaluating the degree of fluency is employed [42]. The dataset is compared with several other existing methods such as the LEAD-3 model, which produces a summary as a baseline with leading 3 sentences in the document and LREG which is a logistic classifier with rich features especially employed as a baseline [2]. The methods Neural Network systems with sentence based extraction (NN-SE) and word based extraction (NN-WE) are trained with the learning rate of 0.001 [13] are used for comparison. The works proposed by the author Nallapati *et al.* [2] such as Classifier based and selector based Recurrent Neural Network architecture proposed by Nallapati *et al.* [2] with shallow and deep models for each architecture and SummaRuNNer, a double layer RNN based sequence classifier that supports extractive and abstractive summary are also compared [18]. The results are presented in **Table 1**.

Table 1 Performance of various models for Daily Mail Corpus.

Models	Recall at 75b			Recall at 275b			Average rank
	Rouge-1	Rouge-2	Rouge-L	Rouge-1	Rouge-2	Rouge-L	
LEAD – 3	21.9	7.2	11.6	40.5	14.9	32.6	6.50
LREG	18.5	6.9	10.2	N/A	N/A	N/A	4.00
NN-SE	22.7	8.5	12.5	**42.2**	**17.3**	34.8	4.00
NN-WE	16.0	6.4	10.2	33.9	10.2	23.5	8.17
Shallow Selector	25.6	10.3	14.0	41.3	16.8	34.9	3.67

Models	Recall at 75b			Recall at 275b			Average rank
	Rouge-1	Rouge-2	Rouge-L	Rouge-1	Rouge-2	Rouge-L	
Deep Selector	26.1	10.7	14.4	41.3	15.3	33.5	3.50
Shallow Classifier	26.0	10.5	14.23	42.1	16.8	34.8	3.00
Deep Classifier	**26.2**	10.7	14.4	**42.2**	16.8	35.0	1.83
SummaRuNNer-abs	23.8	9.6	13.3	40.4	15.5	32.0	5.50
SummaRuNNer	**26.2**	**10.8**	14.4	42.0	16.9	34.1	2.33
Proposed Model	26.1	**10.8**	**14.5**	42.1	16.8	**35.2**	1.67

The results of the proposed model are in par with the complex neural network based summarization. In general, the proposed method delivers improved results than 8 existing models out of 10 models under comparison. On evaluating the potential of the proposed model, the average rank is computed based on the obtained results. The proposed model is having approximately 1.67 of average ranks which is a top-ranked among other existing models.

Results on CNN/Daily Mail Corpus

The performance of the proposed model is also evaluated on CNN/Daily Mail dataset using a full-length Rouge-Recall metric with unigram overlap (ROUGE-1), bigram overlap (ROUGE-2) and longest common sequence (ROUGE-L) that are common between the obtained summary and the summary in the reference. The resulting values are presented in **Table 2**.

Table 2 Performance of various models for CNN/Daily Mail Corpus.

Models	Full-Length F_1 metric			Average rank
	ROUGE-1	ROUGE-2	ROUGE-L	
words-lvt2k	32.49	11.84	29.47	10.33
words-lvt2k-hieratt	32.75	12.21	29.01	10.00
lead-3 baseline	39.2	15.7	35.5	3.67
Attentional RNN	35.46	13.30	32.65	8.67
SummaRuNNer-abs	37.5	14.5	33.4	6.33
SummaRuNNer	**39.6**	16.2	35.3	2.67
seq-to-seq + attn baseline (150k vocab)	30.49	11.17	28.08	12.33
seq-to-seq + attn baseline (50k vocab)	31.33	11.81	28.83	12.00
pointer-generator	36.44	15.66	33.42	5.67
pointer-generator + coverage	39.53	**17.28**	36.38	1.67
Graph based	38.1	13.9	34.0	15.33
Multitask Learning	35.8	13.6	33.4	7.33
Proposed Model	39.34	17.21	**36.5**	2.00

The results of flat attention models such as words-lvt2k and words-lvt2k-hieratt [17] are analyzed. Other models such as LEAD-3 model that produces the summary with leading 3 sentences as the baseline, abstractive summarization using attentional RNN [1], SummaRuNNer, a double layer RNN based sequence classifier that supports both extractive and abstractive summary are also compared [18]. Other methods proposed by See *et al.* [19] such as sequence-to-sequence attention model by varying the baseline to 150k vocabulary and 50k vocabulary, pointer generator model, and pointer network model with coverage mechanism are also used for evaluation. The results of the methods such as graph based [20] and multitask learning [21] are compared with the proposed model.

On analyzing the results presented in **Table 2**, the proposed model outperforms than 10 existing models out of 12 models for ROUGE-1 metrics. The proposed model is better than other existing models in the case of ROUGE-2 and ROUGE-L metrics. To identify the efficiency of the model, the average rank is computed for all the existing and proposed model in which, the pointer generator with coverage has acquired the score of 1.53 and is at first position whereas the proposed model is at the second position with the score of 2.00 among thirteen models under comparison.

Results on DUC 2004

Based on the literature survey made with the existing models, the proposed model is also evaluated on DUC 2004 dataset and the results are compared with the existing model using ROUGE metric at 75 bytes of summary with unigram overlap (ROUGE-1), bigram overlap (ROUGE-2) and longest common sequence that are common between the obtained summary and the summary in the reference (ROUGE-L). The results for various models are presented in **Table 3**.

Table 3 Performance of various models for DUC 2004 dataset.

Models	Recall at 75 bytes			Average rank
	ROUGE-1	ROUGE-2	ROUGE-L	
IR	11.06	1.67	9.67	12.67
COMPRESS	19.77	4.02	17.30	11.33
PREFIX	22.43	6.49	9.67	10.33
W & L	22	6	17	11.00
TOPIARY	25.12	6.46	20.12	9.33
Moses+	26.50	8.13	22.85	7.67
ABS	26.55	7.06	22.05	7.33
ABS+	28.18	8.49	23.81	5.33
Reference	**29.12**	8.38	24.46	3.33
RAS-Elman	**28.97**	8.26	24.06	4.33
words-lvt2k-1sent	28.35	**9.46**	24.59	3.00
words-lvt5k-1sent	28.61	9.42	25.24	3.00
Proposed Model	28.81	9.43	**25.31**	2.00

The baselines such as IR [43], COMPRESS model, a sentence compression baseline [44], PREFIX that returns the first 75 characters of the input, TOPIARY that employs both compression system and unsupervised topic detection [22], W & L system, a quasi-synchronous grammar approach [9], MOSES+, a phrase based statistical model for summary extraction [23], attention based model for abstractive summary extraction such as ABS and ABS+ [3], along with the reference that uses a set of manual summary extraction are used for result evaluation and all these values of the results are presented by Rush *et al.* [3]. Additionally, as per the results reported by Nallapati *et al.* [1], RAS-Elam model [17], words-lvt2k-1sent, a model trained for the first sentence, words-lvt5k-1sent, a model with a larger vocabulary of size 5k [1] are also compared with the proposed model.

On analyzing the results presented in **Table 3**, the proposed model outperforms than 10 existing models out of 12 models for ROUGE-1 metric, 11 out of 12 existing models for ROUGE-2, and provides better results for ROUGE-L metric. To identify the efficiency of the model, the average rank is computed for all the existing and proposed models and is given in the last column in **Table 3**, from which it is clear that the proposed model has acquired a score of 2.00 and is at the first position.

Results on Opinosis dataset

Opinosis is a publically available dataset that is not widely used. Kågebäck *et al.* [11] employed this dataset for the performance evaluation of the extracted document summarization. Based on their report, the proposed model is also evaluated on the Opinosis dataset and the results are compared with the few existing models using ROUGE metrics that compare the words in the generated summaries with that of

the referenced one. The precision, recall, and F-scores are averaged for the ROUGE metric with unigram overlap (ROUGE-1), bigram overlap (ROUGE-2), and skip bigrams that allows 4 terms between the 2 compared terms (ROUGE-SU4) are analyzed for the proposed and existing models. Kågebäck *et al.* [11] analyzed the submodular optimization model (SMO) suggested by Lin and Bilmes [26] by varying the combinations of word embedding methods such as CW vectors [27], Continuous Skip-gram denoted as Word2Vec (W2V), phrase embedding methods such as vector addition (Add), Unfolding Recursive Auto-encoder (RAE) [10] and similarity methods such as Cosine Similarity (Cos) and Euclidean distance (Eud). The authors also reported the results which are used for comparison of the proposed model. The values are listed in **Table 4**.

Table 4 Performance of various models for Opinosis dataset.

Recall				
Models	**ROUGE-1**	**ROUGE-2**	**ROUGE-SU4**	**Average rank**
SMO	25.82	3.92	9.15	7.67
CW_RAE_Cos	27.37	4.68	9.61	6.67
CW_RAE_Euc	29.25	4.82	9.95	4.67
CW_Add_Cos	**34.72**	**5.89**	12.38	1.33
CW_Add_Euc	29.12	5.12	10.54	4.33
W2V_Add_Cos	30.86	5.71	11.94	3.00
W2V_Add_Euc	28.71	3.86	9.78	6.67
Proposed Model	34.51	5.77	**12.41**	1.67
Precision				
SMO	19.58	2.50	6.74	4.67
CW_RAE_Cos	19.89	3.18	6.23	3.67
CW_RAE_Euc	19.77	3.24	6.17	4.00
CW_Add_Cos	11.75	1.81	3.27	8.00
CW_Add_Euc	**22.75**	**3.60**	7.59	1.33
W2V_Add_Cos	16.81	3.08	5.52	5.67
W2V_Add_Euc	16.67	1.95	4.69	7.00
Proposed Model	22.17	3.57	**7.84**	1.67
F-Score				
SMO	20.57	2.87	6.73	6.33
CW_RAE_Cos	22.00	3.58	6.95	4.67
CW_RAE_Euc	22.62	3.67	7.04	3.67
CW_Add_Cos	17.16	2.71	5.03	7.67
CW_Add_Euc	**24.88**	**4.10**	8.35	1.33
W2V_Add_Cos	20.93	3.82	7.12	3.67
W2V_Add_Euc	20.75	2.54	6.15	7.00
Proposed Model	24.73	3.92	**8.76**	1.67

On analyzing the results reported in **Table 4**, with recall, precision, and F-score metrics, the proposed model outperforms than existing models at ROUGE-SU4. However, for ROUGE-1 and ROUGE-2, the proposed model provides good results than most of the existing methods. To identify the efficiency of the model, the average rank is computed for all the models under compassion, in which the proposed model has acquired a score of 1.67 and is at the second top position.

From the experimental and result analysis made, it is clear that the proposed model provides better performance than most of the existing methods. Despite complex neural network based models, the main

advantage of the proposed model is its simplicity. Though the model is not suitable for abstractive summary extraction, it is good at producing extractive summary from the single document.

Conclusions

In this paper, the automated single document summarization by constructing similitude graphs from the extracted text segments was suggested for extractive summarization. The text segments were extracted and the feature was computed for all the segments based on 3 feature extraction methods. Based on the computed features the similarity between the segments was evaluated to construct the graph. The segments were ranked for including the segments in the extractive summary by computing the graph score and the sentence segment score. The experimental analysis was performed to analyze the performance of the proposed system using the variants of ROUGE metrics with 4 popular datasets in the document summarization domain. The results show that the proposed model provides good results that are better than several existing summarization models. There are several directions available for future work. The method produces an extractive summary in which the method can be extended to support an abstractive summary. Also, the method provides better results for single document summarization in which it can be extended to multiple document summarization. The method can be improved and modified in such a way to achieve 100 % quality results than all the other models which are compromised with a few existing models. The future scope of the work aims at extending the model that suits for other data types such as video and image as the current work supports text data.

References

[1] R Nallapati, B Zhou, C Gulcehre, CND Santos and B Xiang. Abstractive text summarization using sequence-to-sequence RNNs and beyond. *In*: Proceedings of the SIGNLL Conference on Computational Natural Language Learning, Berlin, Germany. 2016, p. 280-90.

[2] R Nallapati, B Zhou and M Ma. Classify or select: Neural architectures for extractive document summarization, Available at: https://arxiv.org/abs/1611.04244, accessed November 2020.

[3] AM Rush, S Chopra and J Weston. A neural attention model for abstractive sentence summarization, Available at: https://arxiv.org/abs/1509.00685, accessed September 2020.

[4] R Mihalcea and P Tarau. A language independent algorithm for single and multiple document summarization. *In*: Proceedings of the International Joint Conference on Natural Language Processing, Jeju Island, Korea. 2005, p. 19-24.

[5] PD Turney. Learning algorithms for keyphrase extraction. *Inform. Retrieval* 2000; **2**, 303-36.

[6] M Litvak and M Last. Graph-based keyword extraction for single-document summarization. *In*: Proceedings of the Workshop Multi-source Multilingual Information Extraction and Summarization, Manchester, United Kingdom. 2008, p. 17-24.

[7] G Erkan and D Radev. Lexpagerank: Prestige in multi-document text summarization. *In*: Proceedings of the Conference on Empirical Methods in Natural Language Processing, Barcelona, Spain. 2004, p. 365-71.

[8] R McDonald. *A study of global inference algorithms in multi-document summarization*. *In*: G Amati, C Carpineto and G Romano (Eds.). Advances in Information Retrieval. Springer, Berlin, Heidelberg, 2001, p. 557-64.

[9] K Woodsend, Y Feng and M Lapata. Generation with quasi-synchronous grammar. *In*: Proceedings of the Conference on Empirical Methods in Natural Language Processing, Massachusetts, United Kingdom. 2010, p. 513-23.

[10] R Socher, EH Huang, J Pennin, CD Manning and AY Ng. Dynamic pooling and unfolding recursive autoencoders for paraphrase detection. *In*: Proceedings of the 24th International Conference on Neural Information Processing Systems, Granada, Spain. 2011, p. 801-9.

[11] M Kågebäck, O Mogren, N Tahmasebi and D Dubhashi. Extractive summarization using continuous vector space models. *In*: Proceedings of the 2nd Workshop on Continuous Vector Space Models and their Compositionality, Gothenburg, Sweden. 2014, p. 31-9.

[12] R Nallapati, B Zhou and B Xiang. Sequence-to-sequence RNNs for text summarization, Available at: https://arxiv.org/pdf/1602.06023v1.pdf, accessed January 2020.

[13] J Cheng and M Lapata. Neural summarization by extracting sentences and words. *In*: Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics, Berlin, Germany. 2016, p. 484-94.

[14] O Sornil and K Gree-Ut. An automatic text summarization approach using content-based and graph-based characteristics. *In*: Proceedings of the IEEE Conference on Cybernetics and Intelligent Systems, Bangkok, Thailand. 2006, p. 1-6.
[15] YA AL-Khassawneh, N Salim and OA Isiaka. Extractive text summarisation using graph triangle counting approach: Proposed method. *In*: Proceedings of the International Conference of Recent Trends in Information and Communication Technologies, Johor, Malaysia. 2014, p. 300-11.
[16] T Mori. Information gain ratio as term weight: the case of summarization of IR results. *In*: Proceedings of the 19th International Conference on Computational Linguistics, Taipei, Taiwan. 2002, p. 1-7.
[17] S Chopra, M Auli and AM Rush. Abstractive sentence summarization with attentive recurrent neural networks. *In*: Proceedings of the Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, California. 2016, p. 93-8.
[18] R Nallapati, F Zhai and B Zhou. Summarunner: A recurrent neural network based sequence model for extractive summarization of documents. *In*: Proceedings of the Thirty-First AAAI Conference on Artificial Intelligence, California. 2017, p. 1684-90
[19] A See, PJ Liu and CD Manning. Get to the point: Summarization with pointer-generator networks. *In*: Proceedings of the Annual Meeting of the Association for Computational Linguistics, Vancouver, Canada. 2017, p. 1073-83.
[20] J Tan, X Wan and J Xiao. Abstractive document summarization with a graph-based attentional neural model. *In*: Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics, Vancouver, Canada. 2017, p. 1171-81.
[21] Y Chen, Y Ma, X, Mao and Q Li. Multi-task learning for abstractive and extractive summarization. *Data Sci. Eng*. 2019; **4**, 14-23.
[22] D Zajic, B Dorr and R Schwartz. Bbn/umd at duc-2004: Topiary. *In*: Proceedings of the HLT-NAACL Document Understanding Workshop, Boston. 2004, p. 112-9
[23] P Koehn, H Hoang, A Birch, C Callison-Burch, M Federico, N Bertoldi, B Cowan, W Shen, C Moran, R Zens and C Dyer. Moses: Open source toolkit for statistical machine translation. *In*: Proceedings of the 45th Annual Meeting of the Association for Computational Linguistics Companion Volume Proceedings of the Demo and Poster Sessions, Prague, Czech Republic. 2007, p. 177-80.
[24] S Wubben, EJ Krahmer and AVD Bosch. Sentence simplification by monolingual machine translation. *In*: Proceedings of the 50th Annual Meeting of the Association for Computational Linguistics, Jeju Island, Korea. 2012, p. 1015-24.
[25] K Ganesan, C Zhai and J Han. Opinosis: A graph based approach to abstractive summarization of highly redundant opinions. *In*: Proceedings of the 23rd International Conference on Computational Linguistics, Beijing, China. 2010, p. 340-8.
[26] H Lin and J Bilmes. A class of submodular functions for document summarization. *In*: Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies, Oregon. 2011, p. 510-20.
[27] R Collobert and J Weston. A unified architecture for natural language processing: Deep neural networks with multitask learning. *In*: Proceedings of the 25th International Conference on Machine Learning, New York. 2008, p. 160-67.
[28] T Mikolov, K Chen, G Corrado and J Dean. Efficient estimation of word representations in vector space, Available at: https://arxiv.org/abs/1301.3781, accessed January 2020.
[29] T Mikolov, I Sutskever, K Chen, GS Corrado and J Dean. Distributed representations of words and phrases and their compositionality. *In*: Proceedings of the Conference on Advances in Neural Information Processing Systems, New York. 2013, p. 3111-9.
[30] JM Torres-Moreno, PL St-Onge, M Gagnon, M El-Beze and P Bellot. Automatic summarization system coupled with a question-answering system (QAAS), Available at: https://arxiv.org/abs/0905.2990, accessed January 2020.
[31] AAL Below. *Information retrieval data structures and algorithms.* Prentice Hall, New Jersey, 1992.
[32] SS Bama, I Ahmed and A Saravanan. A mathematical approach for improving the performance of the search engine through web content mining. *J. Theor. Appl. Inform. Tech*. 2014; **60**, 343-50.
[33] SS Bama, MI Ahmed and A Saravanan. A mathematical approach for mining web content outliers using term frequency ranking. *Indian J. Sci. Tech.* 2015; **8**, 1-5.
[34] MF Porter. An algorithm for suffix stripping. *Program Electron. Libr. Inform. Syst.* 1980; **14**, 130-7.

[35] SS Bama, MSI Ahmed and A Saravanan. *Relevance re-ranking through proximity based term frequency model. In*: G Stojanov and A Kulakov (Eds.). Advances in Intelligent Systems and Computing. Springer, Cham, 2016, p. 216-29.
[36] SS Bama, MI Ahmed and A Saravanan. Enhancing the search engine results through web content ranking. *Int. J. Appl. Eng. Res.* 2015; **10**, 13625-35.
[37] S Bertazzon and S Olson. Alternative distance metrics for enhanced reliability of spatial regression analysis of health data. *In*: Proceedings of the International Conference on Computational Science and Its Applications, Berlin, Germany. 2008, p. 361-74.
[38] R Shahid, S Bertazzon, ML Knudtson and WA Ghali. Comparison of distance measures in spatial analytical modeling for health service planning. *BMC Health Serv. Res.* 2009; **9**, 200.
[39] SS Bama and A Saravanan. Efficient classification using average weighted pattern score with attribute rank based feature selection. *Int. J. Intell. Syst. Appl.* 2019; **11**, 29-42.
[40] KM Hermann, T Kocisky, E Grefenstette, L Espeholt, W Kay, M Suleyman and P Blunsom. Teaching machines to read and comprehend. *In*: Proceedings of the 28th International Conference on Neural Information Processing Systems, Massachusetts. 2016, p. 1693-701.
[41] F Dernoncourt, M Ghassemi and W Chang. A repository of corpora for summarization. *In*: Proceedings of the 11th International Conference on Language Resources and Evaluation, Miyazaki, Japan. 2018, p. 3221-7.
[42] CY Lin. Rouge: A package for automatic evaluation of summaries. *In*: Proceedings of the Workshop on Text Summarization Branches out, Barcelona, Spain. 2004. p. 74-81.
[43] H Schütze, CD Manning and P Raghavan. *An introduction to information retrieval.* Vol 39. Cambridge University Press, Cambridge, England, 2009.
[44] J Clarke and M Lapata. Global inference for sentence compression: An integer linear programming approach. *J. Artif. Intell. Res*. 2008; **31**, 399-429.