Expert Systems With Applications 143 (2020) 112958

Contents lists available at ScienceDirect

Expert Systems With Applications

journal homepage: www.elsevier.com/locate/eswa

Text document summarization using word embedding

Mudasir Mohd [a,b,*], Rafiya Jan [a], Muzaffar Shah [b]

[a] South Campus, University of Kashmir, India
[b] Datoin, Bangalore, India

ARTICLE INFO

Article history:
Received 15 June 2019
Revised 15 September 2019
Accepted 16 September 2019
Available online 3 October 2019

Keywords:
Extractive summarization
Semantic summarization
Distributional semantics
Text summarization

ABSTRACT

Automatic text summarization essentially condenses a long document into a shorter format while preserving its information content and overall meaning. It is a potential solution to the information overload. Several automatic summarizers exist in the literature capable of producing high-quality summaries, but they do not focus on preserving the underlying meaning and semantics of the text. In this paper, we capture and preserve the semantics of text as the fundamental feature for summarizing a document. We propose an automatic summarizer using the distributional semantic model to capture semantics for producing high-quality summaries. We evaluated our summarizer using ROUGE on DUC-2007 dataset and compare our results with other four different state-of-the-art summarizers. Our system outperforms the other reference summarizers leading us to the conclusion that usage of semantic as a feature for text summarization provides improved results and helps to further reduce redundancies from the input source.

1. Introduction

Text analytics helps in taking insights from the text. Text processing and automatic text summarization are the key tasks in this area. Text summarization identifies and extracts key sentences from the input documents to produce the automatic text summaries of the input documents, and hence is a potential solution to the information overload problem. Moreover, text summaries are used for reducing the length of the input documents without compromising with its overall meaning and information content. Hence, text summarization is the data reduction process for quick consumption by the user.

Automatic text summarization is useful in many ways: Automatic summaries are less biased as compared to human-generated summaries; summaries are pivotal to the question-answering systems; indexing is effective if automatic summarizers are used; reading a summary instead of document reduces reading time; summaries make selection of articles easier during researching; commercial abstracting services employ automatic summarizers to effectively increase their throughput by producing large number of summaries in a shorter time. Hence, we find that there is an excellent motivation to work on the text summarization and improve the process.

Several types of summaries are inferred from a text viz *Extractive Summary*, *Abstractive Summary*, and *Hybrid Summary*. Extractive summarization employs sentence extraction. Summarization is done by the identification of the sentences from the text of the input document. The selected subset is the most important set of sentences to that document and hence the summary. The selection of subset is premised on statistical attributes of the given document that includes frequency of words or phrases, sentence position in the text or topic phrases that are indicatives of important sentences in the text. (Lloret, Romá-Ferri, & Palomar, 2013) designed a text summarization system known as *COMPENDIUM*, used for the summarization of biomedical papers. The system generates the abstracts of biomedical papers. *COMPENDIUM* produces both the abstractive and extractive summaries and the evaluation of summaries is done on both quantitative as well as qualitative aspects. (Rotem, 2003), (Sevilla, Fernández-Isabel, & Díaz, 2016) are two extractive summarizers that produce quality extractive summaries.

Abstract summaries are produced by re-generating the retrieved content. Most of the research work is based on selection and extraction of extracts from the input document based on statistical rules such as the position of a sentence in the document, frequency of a word, etc. The drawback of this approach is that it greatly depends on the format of the text and completely ignores the underlying meaning and semantics of the text. Moreover, it depends on hand-tagged rules which are not flexible and are time-consuming. (Barros, Lloret, Saquete, & Navarro-Colorado, 2019) use semantic

* Corresponding author.
E-mail addresses: mudasir.mohammad@kashmiruniversity.ac.in (M. Mohd), muzaffarshah@datoin.com (M. Shah).

https://doi.org/10.1016/j.eswa.2019.112958
0957-4174/

graph for the production of excellent abstract summaries. Hybrid summarizers are the amalgam of the extractive and abstractive summarizers.

Automatic summarizers summarize text using two approaches namely supervised learning and unsupervised learning. While the supervised learning algorithms are producing excellent results for automatic text summaries, but require abundant labelled training data that is expensive and difficult to create. On the other hand, unsupervised algorithms exploit the statistical and linguistic features to obtain a summary of the input text document. The usage of these statistical and linguistic features tend to miss out the other essential and fundamental aspects of the input text as they tend to focus on the specific inherent properties of the input text documents such as sentence length, sentence position, frequently occurring words in the text, words related to the topic of the text.

The primary disadvantage of the automatic summarization systems is that they focus on the statistical features while ignoring the semantic meaning of the input document. In this proposed approach, we focus on the semantic information contained in the sentences and propose an automatic summarizer that uses the semantics of the text as a feature in addition to the other statistical features. We use distributional semantic models to capture the semantics of the text. The distributional semantic model works on the principle of distribution hypothesis which states that words occurring in the same context convey the same meanings. Thus, we use the distributional semantic models to compute the semantic coherence and use it as feature in the text summarization process. Our experimental evaluation shows that the semantics improve the text summarization process and thus produce quality summaries that do not ignore the meaning of the text that it wants to convey to its readers. The characteristic of the proposed model is to employ the distributional semantic model to capture the semantics of the text for summarization task is exciting and novel considering that no summarization work done earlier has focused on this aspect of the automatic summarization.

Our proposed system consists of four steps: (1) We use semantics of the text as a feature, captured using distributional semantic model, i.e., *Word2Vec* (Mikolov, Chen, Corrado, & Dean, 2013) model. More precisely, each sentence is transformed into a semantically rich big-vector (BV) - a semantic extension of the sentence which is obtained using *Word2Vec*. All the words of the sentence are feed to the *Word2Vec* model and retrieved word vectors are combined into big-vector; (2) *k*-means clustering (Hartigan & Wong, 1979) algorithm is used for clustering the big-vectors into different clusters;(3) ranking algorithm is used on each cluster to identify semantically essential sentences for creating a generic extractive summary. We propose a ranking algorithm for ranking essential sentences from each cluster using the method of sentence scoring to create extractive summaries.

We performed the experiments using our proposed automatic text summarizer on DUC-2007 dataset[1] and evaluated the results using ROUGE (Recall-Oriented Understudy for Gisting Evaluation) metrics. ROUGE performs the comparison of system generated summaries with the benchmark summaries also know as reference summaries that are typically human-produced. *Recall*, *precision* and *F-score* were obtained for four different types of ROUGE, i.e. ROUGE-1, ROUGE-2, ROUGE-L, and ROUGE-SU4. We also performed a comparative analysis with five other state-of-art text summarizer existing in the literature to check the competitive efficacy of our proposed summarizer and found that the use of semantics as a feature significantly improves the summarizer performance.

[1] https://duc.nist.gov/duc2007/.

The proposed automatic summarizer is built keeping in view that the semantics of text is to be captured which lacked in the previous state-of-art conducted in the area of automatic text summarizers. Moreover, the results of the proposed automatic summarizer confirm the competitive effectiveness of the proposed automatic text summarizer and the usage of semantics as feature to improve the results of the unsupervised extractive summarizer.

Rest of the paper is organised into the following sections: Section 2 provides detailed state-of-the-art techniques used in the literature, Section 3 discuses the proposed methodology. In Section 4 we discuss the experimental setup and the results. Finally, we conclude the paper in Section 5.

2. Related works

This section discusses the different aspects of text summarization, the approaches used by automatic text summarizers and their limitations and relative advantages over the other. The aim of this exhaustive review of techniques, approaches, and use of feature selection was to obtain conclusions and determine pending issues.

2.1. Unsupervised text summarization

Luhn started the initial and groundbreaking work in the field of automatic text summarization as early as the 1950s. Luhn concluded with the hypothesis that more frequent words are directly proportional to the importance of words in the document. Thus higher the frequency of the word, higher is its importance in that document. He further eluded that more frequent words are either topic words or descriptive words, and hence the sentences that contain them are also important and therefore should be included in the summary (Luhn, 1958). Edmundson and Wyllys (1961) extended Luhn's work by including several features that increase the score of candidate sentences for text summarization. Features used by them to assign ranks to the sentences in the documents are: (1) Frequency of words in the document; (2) number of tittle word occurrences in a sentence; (3) Position of sentence in document, higher the sentence in the document higher is its importance; (4) count of cue-Phrases like "in short, in summary". Baxendale (1958) laid the foundation of abstract summarizers. He introduced the idea of language generators in the field of automatic text summarization. More precisely, they stressed that summaries could also be produced by including those sentences in the summaries that are not present in the original document and hence gave the concept of abstract summarizers.

Various other early works have used single document extractive summarization that employs the statistical techniques as proposed by Edmundson like (Afantenos, Karkaletsis, & Stamatopoulos, 2005) use ranking algorithms for sentence scoring. (Gong & Liu, 2001) exploited Latent Semantic Analysis to find the correlation between the topics and sentences to find the effective summary of the document. (Mihalcea & Tarau, 2004) proposed TextRank as an algorithm to find extractive summaries of documents where they used the sum of weighted similarities between sentences.

2.2. Query based text summarization

Query based summarization is another well-established approach in the automatic text summarization task. Using this approach the summary of the text document is computed on the basis of an input query. More precisely, this method is centering around the user query parameters and hence centred around the user queries provided to the system by users.

In query-based summarization techniques input text is probed for the query terms provided by the user and the query phrases

sentences are given higher ranks as compared to the sentences with words of single query. The output for the summary is higher scored sentences along with the other structural components. There may be some extracts from different portions or sections of the input text and hence result of query-based summarizers is the union of all the extracts (Pembe & Güngör, 2007).

(Zhao, Wu, & Huang, 2009) used a query expansion method along with the graph base summarization technique to resolve the problem occurring due to the limited information in the original queries. They used the document set on which the summarization is performed for the expansion of queries rather than the external sources like WordNet as used by (Li, Li, Li, Chen, & Wu, 2005). They explored Sentence-to-Sentence and Sentence-to-Word relationships on the document set for the query expansion. Experiments performed on DUC 2005, and 2006 datasets led them to the conclusion that query expansion produced better results than the systems without query expansion. (Ouyang, Li, Li, & Lu, 2011) used Support Vector Regression (SVR) for calculating the importance of sentence in a document for its inclusion in the summary in response to the user-specified query. (Carbonell & Goldstein, 1998) used Maximal Marginal Relevance (MMR) for combining query-relevance with the information-novelty for text summarization task.

2.3. Machine learning based approaches

With the advances in machine learning approaches, in the recent years Machine Learning models are being utilized for generating extractive summaries. These approaches use relevant features which are chosen manually to train supervised learning algorithms to create a model. The model is then used for classification to determine whether to include a sentence in summary or not. Examples include, (Wong, Wu, & Li, 2008) used Support Vector Machines (SVM) and Naive Bayes model with features like relevance, topic, event and content for generating extractive summaries. (Neto, Freitas, & Kaestner, 2002) used statistical and linguistic features directly extracted from the original textual document and employed machine learning techniques to build an extractive summarizer. (Rabiner & Juang, 1986) used hidden Morkov chains (HMMs) and (Conroy & O'leary, 2001) to improve the extractive summary results by ordering the sentence in the documents.

2.4. Neural networks based approaches

With the advances in deep learning and a decrease in the cost of computer memory neural network based automatic summarization has picked up the pace. In contrast to the traditional automatic summarizers, the neural network based summarizers are achieving better results with less human involvement only if sufficient training data is available (Dong, 2018). In 2014 (Kågebäck, Mogren, Tahmasebi, & Dubhashi, 2014) used neural based continuous vectors for extractive summary and results were consistent and thus paved the way for using neural networks for text summarization tasks. This research marked the beginning of using neural networks for the automatic text summarization task, which provided good results compared to the traditional supervised and unsupervised automatic summarization systems due to the superior performance of the artificial neural networks. Followed by this work (Rush, Chopra, & Weston, 2015) was the one who first proposed an abstractive summarizer that used Convolutional Neural Networks (CNN) to generate the text summaries. They used CNN as an encoder to generate the extractive summary and then used Neural networks as a decoder to transform the extractive summary into the abstractive summary. (Chopra, Auli, & Rush, 2016) proposed an abstractive summarizer which is an extension of the abstractive summarizer proposed by (Rush et al., 2015). They used Conditional Recurrent Neural Networks (RNN) to obtain the abstractive summary of the input document. They have used a novel mechanism consisting of attenuation based convolutional encoder to obtain the extractive summary and then feed the same to the decoder to obtain the abstractive summary. They interlinked their encoder with the decoder by providing conditioning information for generating new words for the sentence based on the score of the input sentence in the form of scores computed over the same sentence. (Nallapati, Zhou, Gulcehre, Xiang et al., 2016) used an attentional encoder-decoder RNN to produce abstract summary of the input text document. They used a generator-pointer model so that the decoder is able to generate words in the source text. (Gu, Lu, Li, & Li, 2016) proposed COPYNET, a sequence-to-sequence learning which copies portions of source text at appropriate intervals in source documents. They used COPYNET with the encoder-decoder structure for text summarization and outperformed the RNN-based text summarization models. (See, Liu, & Manning, 2017) used pointer-generator model to address the off vocabulary problem in the summarization task utilizing neural networks. (Hu, Chen, & Zhu, 2015) created an extensive and detailed Chinese social media corpus for summarization task. (Chen, Zhu, Ling, Wei, & Jiang, 2016) proposed distraction based neural network model which allows them to distract between the different sections of the input text rather than focus on the specific continuous sections proposed by other similar approaches. They achieved comparable results particularly when the input documents are particularly long. (Ma et al., 2018) introduced a semantic relevance based neural network model that produces semantically relevant summaries of Chinese input text document. (Paulus, Xiong, & Socher, 2017) uses a bi-directional LSTM encoder for extractive summarization. They update the parameters for their model using stochastic gradient descent method and reinforcement learning to update the parameters for their model learning.

2.5. Graph and network-based summaries

(Mao, Yang, Huang, Liu, & Li, 2019) proposed graph-based techniques along with supervised learning and unsupervised learning to produce extractive single-document summaries. Their purpose was to measure the importance of sentences by exploring statistical features of sentences computed using supervised and unsupervised algorithms and then measuring the relationship between sentences using graph methods. They used three techniques in their study. (Amancio, Silva, & da F. Costa, 2015) uses the symmetry of word adjacency networks for the authorship attribution task. They used text as a graph for identifying the author and found 82.5% accuracy when tested on a dataset comprising books written by eight authors. (Tohalino & Amancio, 2018) use multilayer graph models for multi-document summarization task. They represent the sentences as nodes and links between the two sentences as edges. The link between the two sentences of two different documents is represented as inter-layer edge and within the same document is represented as intra-layer. Hence, by using this distinction in a multi-layer graph, they have been able to improve the quality of the generated summaries.

Main drawback of the neural network and machine learning based approach discussed above is that although they are producing good quality summaries but require abundant training data and training time. Unsupervised approaches work well in the generation of the automatic summaries, but the essential features are missing in all the existing studies that are done in the field so far; that is the underlying meaning and semantics of the text are missing. Thus we propose a text summarizer using unsupervised and supervised learning algorithms which uses semantics as a feature

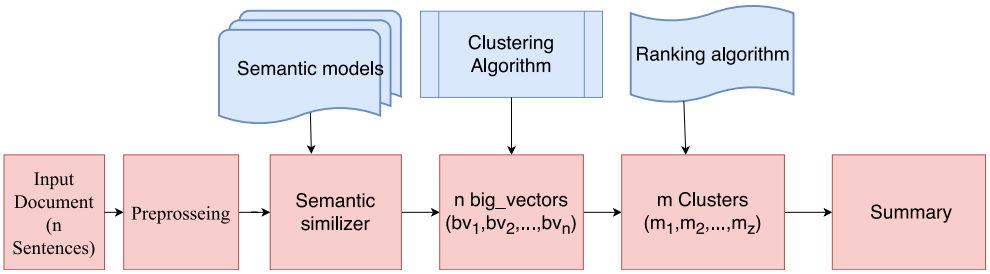

Fig. 1. Overall working of the system.

to summarize the text documents. Our evaluation of results confirms that using semantics as a feature improves system performance as a whole.

3. Methodology

In this section, we describe our model for automatic summarization. The proposed model exploits the semantics of text as a feature along with other stylistic and statistical features. Fig. 1 shows the architecture of our model. Precisely, our proposed approach consists of (1) Perform preprocessing of text for text normalization and remove inconsistencies; (2) capture semantics of text using Distributional Semantic Models; (3) use clustering to group the semantically similar sentence into common clusters; (4) use the ranking algorithm to obtain scores for each sentence from each cluster and (6) normalize the scores for effective sentence extraction to obtain the summary.

3.1. Preprocessing

Preprocessing removes inconsistencies in data and thus produces normalized data. Preprocessing is the first and primary step of our system. We feed the data into our summarizer which is then preprocessed and converted into the format suitable for the summarizing algorithm for processing.It has following steps;

- **Removing URL's:** URLs present in the input document are stripped off by the preprocessing module.
- **Lower Case:** The case of all text in the input document is changed to lowercase by our preprocessing module.
- **Stop-word Removal:** Stop-words are meaningless for the summarization task, thus removed. Stop-words are removed using the Stanford Core NLP package.
- **Tokenization:** Each sentence in the input document is tokenized into words before being processed further. Sentence is broken into words during the tokenization for processing. Tokenization was done using Stanford Core NLP (Manning et al., 2014).
- **Lemmatization:** Next words are reduced to their stems using Stanford core NLP package (Manning et al., 2014).

3.2. Capturing semantics using distributional semantic models

To capture the semantics of text, we employ Distributional Semantic Algorithms as they are the most generic and do not require any lexical and linguistic analysis. Also, these models are independent of any external sources for obtaining semantic information base.

We employ distributional semantic models to capture the semantic coherence between two textual elements. These models are based on the distributional hypothesis. The distributional hypothesis states that the words occurring in the same context often have the same meaning, and thus, we can infer the meaning of them based on their usage. Moreover, the distributional models construct the semantic embeddings using statistical computations of the contexts for the word occurrences. Hence for each word, high-dimensional real-valued vectors are computed and represented as word embeddings or word-vectors. These representations along with their geometric properties in higher-dimensional vector spaces are found to be syntactically and semantically useful to find the coherence of different word usages (Mikolov et al., 2013), (Pennington, Socher, & Manning, 2014), and hence leading to derivations that words that are close in high dimensional vector spaces are syntactically and semantically similar.

To exploit distribution similarity, we use Word2Vec (Mikolov et al., 2013) which is a distributional semantic model that captures the semantics. The employment of Word2vec for capturing semantic similarity of text is considered useful and suitable as it has been used in various methods (Canales, Strapparava, Boldrini, & Martínez-Barco, 2017), (Schütze, 1998), (Barzilay & Lapata, 2008). (Canales et al., 2017) used Word2Vec for bootstrapping to create automatically annotated emotional corpus; (Schütze, 1998) used it in resolving the sense of words in word sense disambiguation and (Barzilay & Lapata, 2008) used it to capture the semantics of text for text coherence problem.

Word2Vec is a two-layer neural network that processes text. It produces as output a set of vectors for the text corpus as input. Vectors produced by Word2Vec are feature vectors of words. Word2Vec algorithm is trained as a vector space representation of terms by exploiting two layers of the neural network. Word2Vec has two architectures: skip-gram and CBOW (continuous bag of words). These architects describe how word embeddings are built by the underneath neural networks for different words. Given the context CBOW predicts the current word and skip-gram predicts the context given the word. We used Skip-gram model in our proposed automatic summarizer. We used the pre-trained model of Word2Vec trained on Google news dataset[2] We introduce the novel approach of measuring semantic similarity using Big-vectors for the text summarization in the next subsection. Big-vectors are also used by (Jan & Khan, 2018) for the purpose of emotion mining.

3.2.1. Big-vector generation

Big-vector of a sentence is created by concatenating the similar words obtained from pre-trained Word2Vec model. Let $\Phi(.)$ be a function that returns a concatenated list of top m similar words of a given word given as $w' = \Phi(w) = w'_1 \oplus w'_2 \oplus, \cdots, \oplus w'_m$. For a sentence with $W = \{w_1, w_2, w_3, \ldots., w_k\}$ as the sequence of k tokenized words, a big-vector BV is populated by concatenating respective top m similar words for each word i.e $BV = \{\Phi(w_1) \oplus \Phi(w_2) \oplus, \cdots, \oplus \Phi(w_k)\}$.

Big-vectors represent rich semantic information of a sentence and are formed on the distribution hypothesis. Specifically, it is se-

[2] https://code.google.com/archive/p/word2vec/.

mantically similar bag-of-word representation of a sentence containing the semantically similar words. We obtain these big vectors of all sentences in a document. Since, the number of words vary in a sentence, thus producing big-vectors of different sizes. To overcome, this problem, fix the size of big-vectors to n-dimensions. The shorter sentences are padded, and for longer sentences, the length is restricted to n.

3.3. Clustering

Once we have semantically enriched representation of sentences in the form of big-vectors, we use clustering algorithms to group the semantically similar sentences and then use ranking algorithm to retrieve and extract top sentences from each group to obtain extractive summary. We use K-means (Hartigan & Wong, 1979) clustering algorithm to form the clusters of the semantically rich big-vectors. We vectorize these big-vectors through $TF-IDF$ and $token-weights$ and pass these vectorized big-vectors as an input to the k-means clustering algorithm. After that clustering is performed and different clusters are formed based on the semantic similarity.

3.4. Ranking algorithm

We then use our novel ranking algorithm to rank the sentences in each cluster for obtaining the extractive summary. Our ranking algorithm uses various statistical features to extract top n sentences from each cluster. The ranking algorithm first obtains the various ranking scores of each sentence in a cluster, and then these scores are normalized, and the final score is the sum of these normalized scores.

The statistical features used by our ranking algorithm are:

- **Sentence length:** (Edmundson & Wyllys, 1961) proposed that the importance of sentence is directly proportional to the length of the sentence. Hence, we use sentence length as a feature for ranking of sentences. Our proposed summarizer uses sentence length as a statistical feature to compute the importance of sentence for ranking purpose. The length of a sentence, $|s_i|$ is given as the number of words after preprocessing.
- **Sentence position:** Edmundson also proposed that more important sentences are at the beginning and at the end of the input text (McCreadie, Macdonald, & Ounis, 2018). Hence, sentence position is an important feature for ranking of the sentences. Baxenlate also suggested that the most important sentences are towards the beginning of the input document (Baxendale, 1958). The sentence position score is calculated as:

$$s_i^p = 1 - \frac{s_i - 1}{|S|}$$

where, s_i^p is the sentence position score of i^{th} sentence of the input document S, and $|S|$ is the number of sentences in the input document.

- **Frequency (TF-IDF):** $TF-IDF$ is an important and characteristic feature of any text summarization system. It identifies the most important terms in any document. The ranking algorithm uses this feature to identify the most important words in a textual document and thus the sentences. The ranking algorithm calculates the $TF-IDF$ of the individual words and then uses the score of individual words to calculate the $TF-IDF$ score of the sentences. Precisely, $TF-IDF$ of the sentence is the summation of the $TF-IDF$ score of the words in the sentence. Formally, $TF-IDF$ of a sentence s_i is calculated as:

$$s_i^{tf} = \sum_{w \in s_i} t_f(w)$$

where the $t_f(w)$ is a function which gives the $TF-IDF$ score of a word w.

- **Noun phrase and verb phrase:** A sentence carrying noun and verb phrase is the most crucial sentence in the input text (Kulkarni & Apte, 2002). A sentence that contains either of the two phrases is an imperative sentence and is ranked higher by our ranking algorithm. Our proposed summarizer uses the Stanford POS tagger (Manning et al., 2014) for the identification of noun and verb phrases in the text and assigns the POS tags to the verb and noun phrases. After the identification of the two, we use NVC (noun verb counter) for each sentence to calculate the number of noun and verb phrases. Higher the NVC count, higher is the rank of the sentence.
- **Proper noun:** Proper nouns include direct references to the subject, and thus the presence of them in the sentence makes the sentence more important (Ferreira et al., 2013). Ranking algorithms assign higher ranks to the sentence with proper nouns in them. Stanford POS tagger is used to extract proper nouns from the input text.
- **Aggregate cosine similarity:** Cosine similarity is used to compute the relatedness between two documents. We use cosine similarity as a feature in our ranking algorithm. We calculate cosine between two sentences. Precisely, higher the cosine similarity between two sentences, higher is the rank of the sentences. The average cosine similarity s_i^c of i^{th} sentence is given as:

$$s_i^c = \frac{\sum_{j=1, j\neq i}^{|S|} c(s_i, s_j)}{|S|}$$

where $c(s_i, s_j)$ gives the cosine similarity between two sentences.

- **Cue-phrases:** Edmundson in 1968 used cue-phrases for the first time for text summarization. Cue phrases are the connecting components between sentences; if a sentence has a

Algorithm 1: Summarizing algorithm.

Result: Summary of input document
Input : Document (d)
Output: Summarized Document (SD)
Let d is the input document having S sentences
let S= $\{s_1, s_2, \ldots s_n\}$ // sentence sequence of d

for all $s_i \in$ S **do**
  $W \leftarrow Tokenization(s_i)$
  where $W = \{w_1, w_2, w_3, \ldots.., w_n\}$

  for all $w_j \in W$ **do**
    $V_j \leftarrow \Phi(W_j)$
  end for
  $BV_i = V_1 \oplus V_2 \oplus \cdots \oplus V_{|W|}$
end for
Let V=$v_1, v_2, \ldots, v_n$ be vectors of BV

for all $v_i \in$ BV **do**
  $\mathbf{BV}_i \leftarrow Vectorize(v_i)$
end for
k_clusters = k_Means_Clustering($\mathbf{BV}_i$)
n_extracts = Ranking (k_clusters)
SD = Join n_extracts

Table 1
Scores of ranking algorithm on example sentences.

#	Sentences	Feature Scores					
		TF-IDF	Token weight	Cosine	Sen-length	Proper-nouns	Total
1	On another occasion, Lewinsky complained that Clinton failed to call even though "the Babba," her name for Hillary Rodham Clinton, was out of town.	0.369	0.71	0.141	0.035	0.025	0.719
2	Placement was great, typeface totally effective and text superlative	0.150	0.96	0.128	0.029	0.005	0.325
3	WASHINGTON They were close friends, so close that the younger woman could confide in the older woman some of her most intimate secrets about a dangerous affair she was having with a married man.	0.341	0.35	0.141	0.043	0.0	0.539
4	And the older woman did not hesitate to pass on some motherly advice.	0.179	0.07	0.122	0.015	0.003	0.316

cue-phrase in the beginning, it implies that it is dependent on the preceding sentence and thus this sentence should be included in the summary (Bhat, Mohd, & Hashmy, 2018; Gupta, Pendluri, & Vats, 2011; Mohd et al., 2016).

The total score of the sentence is then calculated by summing the individual normalized scores of each sentence. Table 1 shows a few sample sentences along with their respective scores.

The words like *moreover*, *however but*, *because*, etc., are called connecting words. If a sentence starts with any of the connecting words it implies its meaning is incomplete without the preceding sentence. Thus if a sentence starting with the connecting word is included in the summary, its preceding sentence is also included in the summary despite of its rank. After ranking algorithm assigns ranks to each sentence in each cluster, the sentences with the best rank are chosen to be a candidates for the extractive summary. Afterwards, the best-scored sentences from each cluster is selected and added to the summary.

Another novel feature of our text summarizer is that it also removes redundancy by checking if the two sentences are written in the rephrased manner, it will eliminate those sentences. Since each cluster is composed of semantically similar sentences and if the ranking score of two sentences is almost similar then it implies both are conveying similar meaning and are thus included only once. This is an exciting and essential feature that will eliminate the semantically similar sentences by including them only once. It is an important feature especially producing summaries of long textual documents wherein authors tend to repeat the sentences by writing them in different ways, and their ranking score will be high, but our proposed system can identify them and discard them from the summaries despite their high ranks. We use sentence position as a metric to chose a sentence among the two semantically similar sentences. Whichever sentence has higher sentence position scores will be selected in summary.

4. Experimental setup and results

This section describes the datasets and the experiments to evaluate the proposed algorithm and presents the recorded results.

4.1. Dataset

In this paper, we have used main task benchmark dataset from Document Understanding Conference[3]~(DUC-2007) collected from ACQUAINT corpus (Vorhees & Graff, 2008). The DUC is series of summarization tasks conducted by National Institute of Standards and Technology (NIST) since 2001, to develop a large scale text corpus for the evaluation of automatic text summarizers. The dataset is comprised of news articles from *New York Associated Press* and *Xinhua News Agency*. The dataset consists of 45 different topics, and each topic has 45 relevant documents associated with it. For each topic, the NIST assessors have created four reference summaries of 250 words approximately, which serves as the ground truth.

4.2. Baselines

We evaluated our proposed approach against the state-of-the-art baselines given as:

- **OPINOSIS** (Ganesan, Zhai, & Han, 2010) is a graph-based summarization framework capable of generating concise and efficient abstractive summaries. The summaries are meaningful and helpful in identifying the critical opinions conveyed by the document. The algorithm uses *word-graph* data-structure to represent the input text, and the graph is repeatedly iterated to obtain the summary.
- **Genism** (Barrios, López, Argerich, & Wachenchauzer, 2016) summarizer is an implementation of the *TextRank* algorithm (Mihalcea & Tarau, 2004). *TextRank* is a graph-based ranking algorithm in which importance of a sentence, represented as a vertex, in the text is computed recursively from the global state of the graph. The algorithm here works by voting, if the vertex is getting linked with some other vertex then the linking vertex gets one vote up and hence more the linkage of the vertex with other vertices; higher the rank of the vertex.
- **PKUSUMSUM** (Zhang, Wang, & Wan, 2016) is a Java summarization platform that supports multiple languages and integrates ten summarization methods. It also supports three summarization tasks that include: Single-document summarization, Multi-document summarization, and Topic-based Multi-document summarization. It integrates stable and various summarization methods also its performance is good enough to be used as a reference system for our evaluation purpose. It provides various summarizing methods such as, *Centroid*, *LexPageRank*, and *TextRank*. For our evaluation, we have used single-document summarizer with *LexPageRank* method for summarization.
- **PyTextRank** is a graph-based summarization method where summaries are produced by employing feature vectors. It is a Python implementation of the variation of *TextRank* algorithm developed by (Mihalcea & Tarau, 2004) that produces text summaries rather than feature vectors. It uses graph algorithms to build the text summaries rather than the obtaining feature vectors from the TextRank algorithm.

All the systems that are used as baselines are state-of-art text summarizing systems and are used in various studies for comparative analysis purpose. These systems employ different algorithms

[3] https://duc.nist.gov/duc2007/tasks.html#main.

Table 2
Averaged summarization results of 25% summary length.

Metric	Rouge Type	Prop. Appr	Gensim	OPINOSIS	PKUSUMSUM	PyTeaser
Pr	ROUGE-1	**0.34**(0.04)	0.05 (0.02)	0.19 (0.17)	0.10 (0.007)	0.030 (0.016)
	ROUGE-2	**0.07**(0.02)	0.02 (0.01)	0.03 (0.05)	0.04 (0.08)	0.097 (0.058)
	ROUGE-L	0.20 (0.03)	0.05 (0.01)	0.05 (0.07)	0.10 (0.01)	**0.44**(0.024)
	ROUGE-SU4	**0.13**(0.02)	0.03 (0.014)	0.09 (0.096)	0.05 (0.007)	0.033 (0.016)
Rc	ROUGE-1	0.34 (0.10)	**0.84**(0.14)	0.07 (0.02)	0.74 (0.05)	0.120 (0.024)
	ROUGE-2	0.08 (0.04)	**0.44**(0.11)	0.01 (0.01)	0.28 (0.06)	0.701 (0.080)
	ROUGE-L	0.20 (0.04)	**0.47**(0.18)	0.05 (0.07)	0.49 (0.04)	0.369 (0.075)
	ROUGE-SU4	0.14 (0.15)	**0.52**(0.11)	0.02 (0.04)	0.39 (0.05)	0.275 (0.0081)
Fs	ROUGE-1	**0.33**(0.06)	0.09 (0.05)	0.08 (0.11)	0.17 (0.01)	0.046 (0.020)
	ROUGE-2	0.07 (0.03)	0.01 (0.01)	0.01 (0.02)	**0.17**(0.02)	0.163 (0.079)
	ROUGE-L	**0.20**(0.03)	0.09 (0.02)	0.07 (0.08)	0.17 (0.02)	0.075 (0.033)
	ROUGE-SU4	**0.13**(0.03)	0.05 (0.01)	0.03 (0.04)	0.09 (0.01)	0.056 (0.023)

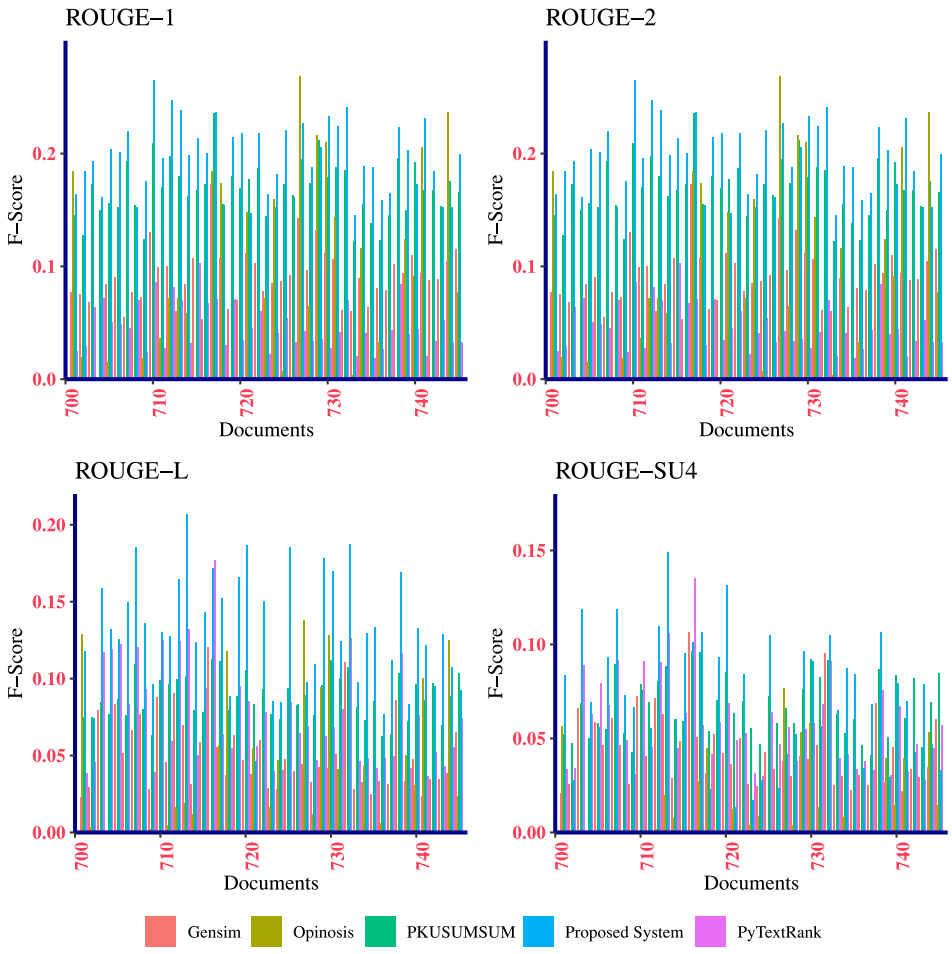

Fig. 2. F-Score for 25% summary length.

for summarizing and thus comparing the performance of our system against them leads to comprehensive and exhaustive comparison against the sate-of-art systems. Moreover, these systems are open source and thus experiments are easily replicable.

4.3. Summarization evaluation

For each topic, we merged its respective 45 documents to obtain a single document. On each topic, we obtained the respective summaries through our approach and the baselines. For extensive evaluation, we generated shorter and lengthy summaries by restricting them to 25% and 50% of the input document respectively.

For evaluation, we use *ROUGE* (Recall-Oriented Understudy for Gisting Evaluation) automatic summarization evaluation toolkit (Ganesan, 2018). It is primarily a set of metrics for evaluating automatic summarization of texts. ROUGE compares summaries at different levels of granularity, and produces four different types of ROUGE metrics namely; ROUGE-1, ROUGE-2, ROUGE-L, and ROUGE-SU4 results. ROUGE-1(2) uses unigrams(bigrams) for measuring coherence between the system summaries and the reference summaries; ROUGE-L uses the summary level Longest Common Sub-sequence (LCS) to match the coherence between the reference and system generated summaries, and ROUGE-SU uses both skip-grams and unigrams for the measurements. The

Table 3
Averaged summarization results of 50% summary length.

Metric	Rouge Type	Prop. Appr	Gensim	OPINOSIS	PKUSUMSUM	PyTextRank
Pr	ROUGE-1	**0.112**(0.014)	0.048 (0.013)	0.064 (0.156)	0.075 (0.013)	0.097 (0.024)
	ROUGE-2	0.140 (0.051)	0.043 (0.027)	0.189 (0.178)	0.049 (0.008)	**0.317**(0.042)
	ROUGE-L	**0.110**(0.016)	0.024 (0.012)	0.088 (0.096)	0.028 (0.005)	0.109 (0.026)
	ROUGE-SU4	**0.104**(0.007)	0.021 (0.009)	0.028 (0.054)	0.024 (0.005)	0.071 (0.023)
Rc	ROUGE-1	0.248 (0.066)	0.521 (0.188)	0.053 (0.075)	**0.649**(0.061)	0.106 (0.021)
	ROUGE-2	0.791 (0.111)	**0.875**(0.066)	0.070 (0.117)	0.850 (0.044)	0.404 (0.052)
	ROUGE-L	0.451 (0.114)	**0.557**(0.088)	0.025 (0.043)	0.508 (0.066)	0.165 (0.035)
	ROUGE-SU4	0.354 (0.012)	**0.476**(0.103)	0.011 (0.019)	0.421 (0.079)	0.095 (0.081)
F1	ROUGE-1	**0.150**(0.015)	0.087 (0.023)	0.067 (0.081)	0.134 (0.022)	0.101 (0.022)
	ROUGE-2	0.230 (0.047)	0.080 (0.045)	0.080 (0.116)	0.092 (0.014)	**0.355**(0.045)
	ROUGE-L	**0.171**(0.010)	0.045 (0.021)	0.029 (0.043)	0.053 (0.010)	0.136 (0.029)
	ROUGE-SU4	**0.153**(0.019)	0.039 (0.171)	0.012 (0.020)	0.045 (0.010)	0.081 (0.027)

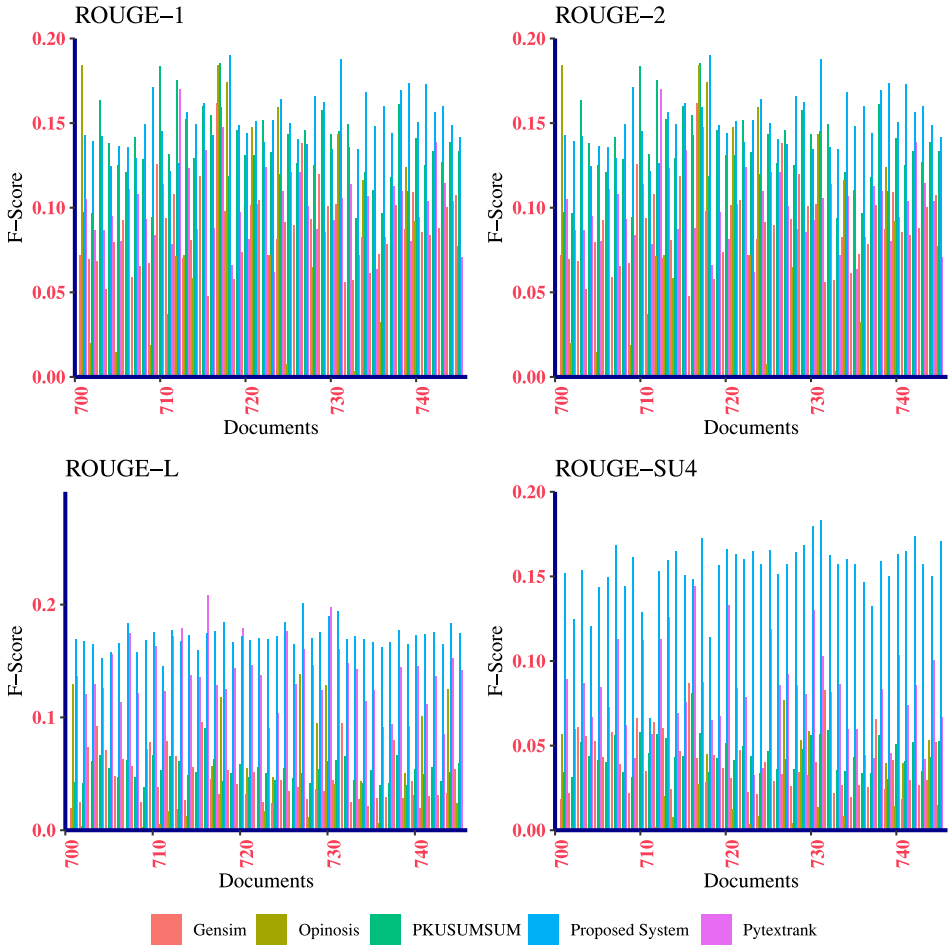

Fig. 3. F-Score for 50% summary length.

summaries obtained through our approach and the baselines are evaluated against ground truth on different ROUGE metrics on *Precision* (Pr),*Recall* (Re), and $F-Score$ (Fs). We computed the average of metrics for all 45 topics and the results for shorter and longer summaries are presented in the Tables 2 and 3 respectively.

As evident from the results of the summarization experiment, our proposed system performs better than the state-of-art-baselines in terms of precision and F-Score. The macro-average of precision values were 34%, 7%, 20% and 13% for ROUGE-1, ROUGE-2, ROUGE-L, and ROUGE-SU4, respectively. Moreover, the macro-averages of F-score values recorded in the 25% summary length are 33%, 7%, 20%, and 13%. Thus results confirm the competitive efficiency of the proposed algorithm.

Furthermore, our proposed method performs better on 50% summary length as shown in Table 3. Our proposed summarizer has F-scores higher than the rest of the system apart from PyTextRank in case of ROUGE-2. With the increase in summary length as expected, recall values go higher but are still lesser than the PKUSUMSUM. Thus we conclude usage of semantic features allows our system to generate better summaries.

As far as the evaluation of the summaries is concerned, the macro-average of the F-scores of different ROUGE metrics is 18.25%, while as those for the other baselines is 6% for Gensim, 5.25% for Opinosis, 15% for PKUSUMSUM and 8% for PyTextRank for 25% summary length. The results confirm the initial hypothesis

that using semantics as a feature for text summarization improves the summary quality.

The results of the evaluation as seen in the Tables 2 and 3 has higher precision scores ROUGE-1, 2, and, SU4 are higher then the rest of the baselines.

The macro-average of the F-scores of our summarizer for 50% summary length is 16.27% while as those of the baselines are 5.75% 4.25% 7.75% and 16.5% for Gensim, Opinosis, PKUSUMSUM and PyTextRank respectively. The performance of summarizer remains constantly good while producing a summary length of 50%.

Higher F-scores of our system are attributed to the usage of semantics as a feature for text summarization system, and thus we conclude that the performance of the system for producing summaries improves with the usage of semantics. Moreover, we achieved better precision and F-scores scores during the evaluation of the system as compared to other baselines. The lower recall of our system is because the system discards some sentences which are statically different but semantically similar.

Figs. 2 and 3 shows the F-Score of the proposed system and the baseline systems used in this paper. It is evident from the figures that our proposed systems produce good quality summaries in comparison to the baselines. Figures show the F-scores of all the documents obtained for different ROUGE metrics.

5. Conclusions and future work

The paper presents a summarization technique based on the distributional hypothesis to capture the semantics of the text for producing better summaries using text summarization process. According to evaluation and comparative analysis, the appropriateness, reliability, and scalability of our summarizer shows that our proposed approach performs better than the baselines. Our main conclusions are: (1) capturing semantics and using it as a feature for summarization helps to improve the precision of our summaries. (2) combining semantic features along with other features tend to produce consistently good summaries. (3) usage of distributional semantic hypothesis tends to produce good results in summarization work as well. The primary disadvantage of the proposed system is that using distributional semantic model is computationally expensive and time consuming.

Our future research will deal with (1) using more than one distributional semantic models for capturing semantics in the text so that semantics are captured at the fine grain level; (2) Improvement of ranking algorithms by exploring more semantic features to be incorporated into our ranking algorithms, as semantic features tend to improve overall system performance; (3) testing the technique on more than one dataset; (4) using BLEU (bilingual evaluation understudy) for evaluation along with the ROUGE.The paper presents a summarization technique based on the distributional hypothesis to capture the semantics of the text for producing better summaries using text summarization process. According to evaluation and comparative analysis, the appropriateness, reliability, and scalability of our summarizer shows that our proposed approach performs better than the baselines. Our main conclusions are: 1

Declaration of Competing Interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

Supplementary material

Supplementary material associated with this article can be found, in the online version, at doi:10.1016/j.eswa.2019.112958.

Credit authorship contribution statement

Mudasir Mohd: Conceptualization, Data curation, Formal analysis, Writing - original draft, Writing - review & editing. **Rafiya Jan:** Conceptualization, Data curation, Formal analysis. **Muzaffar Shah:** Conceptualization, Formal analysis, Writing - original draft.

References

Afantenos, S., Karkaletsis, V., & Stamatopoulos, P. (2005). Summarization from medical documents: A survey. *Artificial Intelligence in Medicine, 33*(2), 157–177.

Amancio, D. R., Silva, F. N., & da F. Costa, L. (2015). Concentric network symmetry grasps authors' styles in word adjacency networks. *EPL (Europhysics Letters), 110*(6), 68001. doi:10.1209/0295-5075/110/68001.

Barrios, F., López, F., Argerich, L., & Wachenchauzer, R. (2016). Variations of the similarity function of textrank for automated summarization. arXiv:1602.03606.

Barros, C., Lloret, E., Saquete, E., & Navarro-Colorado, B. (2019). Natsum: Narrative abstractive summarization through cross-document timeline generation. *Information Processing & Management, 56*(5), 1775–1793.

Barzilay, R., & Lapata, M. (2008). Modeling local coherence: An entity-based approach. *Computational Linguistics, 34*(1), 1–34.

Baxendale, P. B. (1958). Machine-made index for technical literature an experiment. *IBM Journal of Research and Development, 2*(4), 354–361.

Bhat, I. K., Mohd, M., & Hashmy, R. (2018). Sumitup: A hybrid single-document text summarizer. In M. Pant, K. Ray, T. K. Sharma, S. Rawat, & A. Bandyopadhyay (Eds.), *Soft computing: Theories and applications* (pp. 619–634). Singapore: Springer Singapore.

Canales, L., Strapparava, C., Boldrini, E., & Martínez-Barco, P. (2017). Intensional learning to efficiently build up automatically annotated emotion corpora. *IEEE Transactions on Affective Computing*.

Carbonell, J., & Goldstein, J. (1998). The use of mmr, diversity-based reranking for reordering documents and producing summaries. In *Proceedings of the 21st annual international acm sigir conference on research and development in information retrieval* (pp. 335–336). ACM.

Chen, Q., Zhu, X., Ling, Z., Wei, S., & Jiang, H. (2016). Distraction-based neural networks for document summarization. arXiv:1610.08462.

Chopra, S., Auli, M., & Rush, A. M. (2016). Abstractive sentence summarization with attentive recurrent neural networks. In *Proceedings of the 2016 conference of the north american chapter of the association for computational linguistics: Human language technologies* (pp. 93–98).

Conroy, J. M., & O'leary, D. P. (2001). Text summarization via hidden Markov models. In *Proceedings of the 24th annual international acm sigir conference on research and development in information retrieval* (pp. 406–407). ACM.

Dong, Y. (2018). A survey on neural network-based summarization methods. arXiv:1804.04589.

Edmundson, H. P., & Wyllys, R. E. (1961). Automatic abstracting and indexing survey and recommendations. *Communications of the ACM, 4*(5), 226–234.

Ferreira, R., de Souza Cabral, L., Lins, R. D., e Silva, G. P., Freitas, F., Cavalcanti, G. D., … Favaro, L. (2013). Assessing sentence scoring techniques for extractive text summarization. *Expert Systems with Applications, 40*(14), 5755–5764.

Ganesan, K. (2018). Rouge 2.0: Updated and improved measures for evaluation of summarization tasks. arXiv:1803.01937.

Ganesan, K., Zhai, C., & Han, J. (2010). Opinosis: A graph-based approach to abstractive summarization of highly redundant opinions. In *Proceedings of the 23rd international conference on computational linguistics* (pp. 340–348). Association for Computational Linguistics.

Gong, Y., & Liu, X. (2001). Generic text summarization using relevance measure and latent semantic analysis. In *Proceedings of the 24th annual international acm SIGIR conference on research and development in information retrieval* (pp. 19–25). ACM.

Gu, J., Lu, Z., Li, H., & Li, V. O. (2016). Incorporating copying mechanism in sequence-to-sequence learning. arXiv:1603.06393.

Gupta, P., Pendluri, V. S., & Vats, I. (2011). Summarizing text by ranking text units according to shallow linguistic features. In *Advanced communication technology (ICACT), 2011 13th international conference on* (pp. 1620–1625). IEEE.

Hartigan, J. A., & Wong, M. A. (1979). Algorithm as 136: A k-means clustering algorithm. *Journal of the Royal Statistical Society. Series C (Applied Statistics), 28*(1), 100–108.

Hu, B., Chen, Q., & Zhu, F. (2015). Lcsts: A large scale chinese short text summarization dataset. arXiv:1506.05865.

Jan, R., & Khan, A. A. (2018). Emotion mining using semantic similarity. *International Journal of Synthetic Emotions (IJSE), 9*(2), 1–22.

Kågebäck, M., Mogren, O., Tahmasebi, N., & Dubhashi, D. (2014). Extractive summarization using continuous vector space models. In *Proceedings of the 2nd workshop on continuous vector space models and their compositionality (CVSC)* (pp. 31–39).

Kulkarni, A. R., & Apte, M. S. (2002). An automatic text summarization using feature terms for relevance measure.

Li, W., Li, W., Li, B., Chen, Q., & Wu, M. (2005). The hong kong polytechnic university at duc 2005. In *Proceedings of document understanding conferences*. Citeseer.

Lloret, E., Romá-Ferri, M. T., & Palomar, M. (2013). Compendium: A text summarization system for generating abstracts of research papers. *Data & Knowledge Engineering, 88*, 164–175.

Luhn, H. P. (1958). The automatic creation of literature abstracts. *IBM Journal of Research and Development, 2*(2), 159–165.
Ma, S., Sun, X., Li, W., Li, S., Li, W., & Ren, X. (2018). Query and output: Generating words by querying distributed word representations for paraphrase generation. In *Proceedings of the 2018 conference of the north american chapter of the association for computational linguistics: Human language technologies, volume 1 (long papers): 1* (pp. 196–206).
Manning, C. D., Surdeanu, M., Bauer, J., Finkel, J., Bethard, S. J., & McClosky, D. (2014). The Stanford CoreNLP natural language processing toolkit. In *Association for computational linguistics (acl) system demonstrations* (pp. 55–60).
Mao, X., Yang, H., Huang, S., Liu, Y., & Li, R. (2019). Extractive summarization using supervised and unsupervised learning. *Expert Systems with Applications, 133*, 173–181.
McCreadie, R., Macdonald, C., & Ounis, I. (2018). Automatic ground truth expansion for timeline evaluation. In *The 41st international acm sigir conference on research & development in information retrieval* (pp. 685–694). ACM.
Mihalcea, R., & Tarau, P. (2004). Textrank: Bringing order into text. In *Proceedings of the 2004 conference on empirical methods in natural language processing*.
Mikolov, T., Chen, K., Corrado, G., & Dean, J. (2013). Efficient estimation of word representations in vector space. arXiv:1301.3781.
Mohd, M., Shah, M. B., Bhat, S. A., Kawa, U. B., Khanday, H. A., Wani, A. H., … Hashmy, R. (2016). Sumdoc: A unified approach for automatic text summarization. In M. Pant, K. Deep, J. C. Bansal, A. Nagar, & K. N. Das (Eds.), *Proceedings of fifth international conference on soft computing for problem solving* (pp. 333–343). Singapore: Springer Singapore.
Nallapati, R., Zhou, B., Gulcehre, C., Xiang, B. et al. (2016). Abstractive text summarization using sequence-to-sequence rnns and beyond. arXiv:1602.06023.
Neto, J. L., Freitas, A. A., & Kaestner, C. A. (2002). Automatic text summarization using a machine learning approach. In *Brazilian symposium on artificial intelligence* (pp. 205–215). Springer.
Ouyang, Y., Li, W., Li, S., & Lu, Q. (2011). Applying regression models to query–focused multi-document summarization. *Information Processing & Management, 47*(2), 227–237.
Paulus, R., Xiong, C., & Socher, R. (2017). A deep reinforced model for abstractive summarization. arXiv:1705.04304.
Pembe, F. C., & Güngör, T. (2007). Automated querybiased and structure-preserving text summarization on web documents. In *Proceedings of the international symposium on innovations in intelligent systems and applications, istanbul*.
Pennington, J., Socher, R., & Manning, C. (2014). Glove: Global vectors for word representation. In *Proceedings of the 2014 conference on empirical methods in natural language processing (emnlp)* (pp. 1532–1543).
Rabiner, L. R., & Juang, B.-H. (1986). An introduction to hidden Markov models. *IEEE ASSP Magazine, 3*(1), 4–16.
Rotem, N. (2003). Open text summarizer (ots). *Retrieved July, 3*(2006), 2006.
Rush, A. M., Chopra, S., & Weston, J. (2015). A neural attention model for abstractive sentence summarization. arXiv:1509.00685.
Schütze, H. (1998). Automatic word sense discrimination. *Computational Linguistics, 24*(1), 97–123.
See, A., Liu, P. J., & Manning, C. D. (2017). Get to the point: Summarization with pointer-generator networks. arXiv:1704.04368.
Sevilla, A. F., Fernández-Isabel, A., & Díaz, A. (2016). Grafeno: Semantic graph extraction and operation. In *2016 eleventh international conference on digital information management (icdim)* (pp. 133–138). IEEE.
Tohalino, J. V., & Amancio, D. R. (2018). Extractive multi-document summarization using multilayer networks. *Physica A: Statistical Mechanics and its Applications, 503*, 526–539.
Vorhees, E., & Graff, D. (2008). *AQUAINT-2 Information-retrieval text: Research collection*. Linguistic Data Consortium.
Wong, K.-F., Wu, M., & Li, W. (2008). Extractive summarization using supervised and semi-supervised learning. In *Proceedings of the 22nd international conference on computational linguistics-volume 1* (pp. 985–992). Association for Computational Linguistics.
Zhang, J., Wang, T., & Wan, X. (2016). Pkusumsum: A java platform for multilingual document summarization. In *Proceedings of coling 2016, the 26th international conference on computational linguistics: System demonstrations* (pp. 287–291).
Zhao, L., Wu, L., & Huang, X. (2009). Using query expansion in graph-based approach for query-focused multi-document summarization. *Information Processing & Management, 45*(1), 35–41.