Latent Semantic Analysis Approach for Document Summarization Based on Word Embeddings

Kamal Al-Sabahi[1], Zhang Zuping[1*], Yang Kang[1]
[1] School of Information Science and Engineering, Central South University
Hunan, Changsha 410083- China
[e-mail: k.alsabahi@csu.edu.cn]
[e-mail : zpzhang@csu.edu.cn]
[e-mail : yk_ahead@csu.edu.cn]
*Corresponding author: Zhang Zuping

Received February 18, 2018; revised June 26, 2018; accepted July 12, 2018; published January 31, 2019

Abstract

Since the amount of information on the internet is growing rapidly, it is not easy for a user to find relevant information for his/her query. To tackle this issue, the researchers are paying much attention to Document Summarization. The key point in any successful document summarizer is a good document representation. The traditional approaches based on word overlapping mostly fail to produce that kind of representation. Word embedding has shown good performance allowing words to match on a semantic level. Naively concatenating word embeddings makes common words dominant which in turn diminish the representation quality. In this paper, we employ word embeddings to improve the weighting schemes for calculating the Latent Semantic Analysis input matrix. Two embedding-based weighting schemes are proposed and then combined to calculate the values of this matrix. They are modified versions of the augment weight and the entropy frequency that combine the strength of traditional weighting schemes and word embedding. The proposed approach is evaluated on three English datasets, DUC 2002, DUC 2004 and Multilingual 2015 Single-document Summarization. Experimental results on the three datasets show that the proposed model achieved competitive performance compared to the state-of-the-art leading to a conclusion that it provides a better document representation and a better document summary as a result.

Keywords: Word Embedding, Augment Weight, Entropy Frequency, Word2Vec, Document Summarization, Latent Semantic analysis

1. Introduction

In the era of information overload, automatic text summarization is needed in which a short representation of the original document is produced. This short representation should retain the essential information in the document without any redundancy [1, 2]. Over the years, the advancement of natural language processing techniques has benefited the text summarization problem [3]. Several approaches have been introduced to solve the problem, ranging from simple position and word-frequency methods to graph-based and machine learning algorithms [4]. The mutual purpose across those techniques is finding a good document representation that enables the machine to determine the essence of the document from the semantic and conceptual standpoints [5]. Some of these techniques are using Latent Semantic Analysis (LSA), where the document is represented as an input matrix A.

Latent Semantic Analysis is an unsupervised algebraic learning algorithm used for Information Retrieval [5]. This algorithm is used to reveal the latent structure of a document using a combination of statistical and algebraic methods. It provides a way to estimate the relations between words, word-document, and document-document in a larger segment of text by association or semantic similarity. These characteristics have sparked a great interest in applying LSA to solve the summarization problem.

The first step in any LSA-based summarization model is building a document representation, the input matrix. The quality of this matrix is crucial to the performance of the algorithm [6, 7]. Earlier unsupervised document representation approaches mostly used frequency-based and centrality-based methods with the assumption that the most important information tends to appear more frequently in the documents compared to the less important detailed descriptions [3]. The performance of these approaches depends on the quality of human feature engineering, which is a very tedious and challenging task. In addition, they need much processing and external resources. Therefore, traditional methods such as a bag of words (BOW) and TF-IDF often fail to yield a good document representation. For example, words contribute to the TF-IDF score only if they match perfectly, which is not the case often. In natural language, humans use different words to describe the same thing, so naively measuring the similarity between words cannot perfectly reflect the real content similarity [8]. Recently, word embedding has successfully allowed words to match on the semantic level [9]. Word embedding methods learn the continuous distributed vector representation of words with neural networks, which can capture the semantic and/or syntactic relationships. The great thing about word embedding is that it does not require prior knowledge of the natural language or external resources of structured semantic information; it just requires a large amount of unlabeled text data used to create the semantic space [10]. The basic idea behind word embedding is that the embedding of each word represents its meaning. The challenge of using word embedding is in choosing a way of describing the distribution of word embeddings across the semantic space. Naively, averaging or summing the vectors often yield poor distribution. We conjuncture that considering other traditional methods beside word embeddings can capture more features of the text and produce a better representation as a result.

1.1. Research Objectives

The main contribution of this work is proposing an unsupervised approach for extractive single-document summarization that combines the strength of word embedding with the

strength of traditional weighting schemes such as Augment Weight (AW) and Entropy Frequency (EF). The ultimate goal is to improve the LSA-based algorithm to solve the automatic text summarization problem. Although word embedding has been applied to summarization task as a part of a language model, to our knowledge, we are the first to use the learned representation of word embedding to enhance the weighting scheme of the LSA input matrix. In summary, the objectives of this work are:

- Propose a novel local weighting scheme for the words in a sentence, which is a modified version of the augment weight with word embeddings (EMBAW), section 3.2.1, b, (1).
- Propose a novel global weighting scheme for words in the document, which is a modified version of the entropy frequency with word embeddings (EMBEF), section 3.2.1, b, (2).
- Compare the ROUGE results of our model and several baselines on the three datasets, section 5.4.

An extensive experiment has been conducted to compare the performance of the proposed model against several baselines on the three datasets. The evaluation results assert that the proposed model outperforms the state-of-the-art models on those datasets.

The rest of the paper is organized as follows. Section 2 presents the related work with much attention to the ones that are comparable to our model. The proposed approach is presented in Section 3. The complexity analysis of the proposed model is introduced in Section 4. The experiments and results are introduced in Section 5. Finally, we conclude and then discuss the future work in Section 6.

2. Related work

Document summarization is a challenging task; however, it is an important and promising NLP application. Trying to boost the performance of summarization models, researchers are utilizing any advancement in NLP to generate better summaries for a document. Most of the current summarization models can be categorized into two main categories, extractive-based and abstractive-based. The extractive-based ones are the most common, in which important words/sentences are extracted from text documents, and then recombined to form a summary [11]. In the following subsections, we introduce some recent related extractive and abstractive works with more attention to the works that are higher relevant to the study of this paper, including LSA based approaches, embedding-based approaches, and deep learning-based ones.

2.1. LSA based models

The basic workflow of most of the LSA based summarization models consists of three main steps, building of the input matrix, Singular Value Decomposition (SVD), and the sentence selection algorithm [7]. Almost all the current models perform the first two steps in the same way, where TF-IDF is used as a common weighting scheme to build the input matrix. They differ in the algorithm used for selecting sentences for the final summary, called the sentence selection algorithm. The most popular sentence selection algorithm, which we follow in this work, is the one proposed by Steinberger and Jezek [12]. In which both $\boldsymbol{V^T}$ and $\boldsymbol{\Sigma}$ matrixes are

used for the process of sentence selection. The sentence vector length is determined by the concepts whose indexes are less than or equal to the number of dimensions in the new space, given as a parameter. The singular values in $\boldsymbol{\Sigma}$ matrix are used to determine the magnitude of the concepts with respect to their correlation with the text. One of the shortcomings of the current LSA-based summarization models is that they use traditional weighting schemes to build the document representation.

Shen et al. [13] proposed a new latent semantic model for information retrieval. They tried to learn the semantic vector representations for queries and documents by incorporating a special convolutional neural network that has a convolution-pooling structure (CLSM) over the word sequences. First, a low-dimensional continuous feature vector is used to present each word in its context such that it captures the contextual information at the word n-gram level. Next, a sentence level feature vector is formed by aggregating the salient semantic concepts. Finally, the sentence vector was fed into a simple feed-forward neural network that performs a non-linear transformation to extract semantic information used to create a continuous vector representation for the whole text.

2.2. Word Embedding-based models

Trying to improve the semantic document representations, researchers used WordNet or a corpus-based measure in a classic bag of words [8]. However, those methods have some issues that limited their performance where they tend to skip hundreds of important details [14]. Moreover, they need a lot of human feature engineering which is a tedious and complex task. Recently, there are several attempts to build sentence-level and document-level semantic information that go beyond the traditional ones. Mikolov et al. [15] introduce word2vec that was a breakthrough in the direction of text representation. Nowadays, word embeddings become at the center of many NLP applications. Furthermore, word embedding substitutes the external semantic knowledge and make human "feature engineering" unnecessary [10]. In the context of text summarization, the challenge is to create sentence and document embeddings from word embeddings. To this end, several approached have been proposed [16-18]. Some of which used the summation of the word vectors from the trained word embeddings to form sentence and document vectors.

Wieting et al. [19] proposed a paraphrastic sentence embedding model based on a large-scale training set of paraphrase pairs. They intend to encode arbitrary word sequence into a vector such that the sequences with a strong paraphrase relationship have high cosine similarity. Word embedding-based document summarization model was introduced by Kobayashi et al. [17]. In which document-level similarity, represented as a set of word embeddings, is used to summarize documents. The negative summation of the nearest neighbor's distance on the embedding distribution is used as a submodular object function. To choose sentences for the summary, they used a modified greedy algorithm. Another word embedding based model proposed by Rossiello et al. [16]. It is a centroid-based model that employs word2vec to find the centroid by summing the embeddings of the top-ranked words which have TF-IDF greater than the topic threshold. The score of each sentence is calculated by the summation of the vectors of its words.

It is worth mentioning that using the word embeddings alone may make the high frequent unimportant words dominant which in turn diminish the representation quality. In this work, we proposed a new local and global weighting schemes that combine the traditional weighting

schemes with word embeddings to improve the performance of LSA on the document summarization task.

2.3. Deep learning-based approaches

The advancement in computational resources and training algorithms have sparked a great interest in applying deep learning techniques to solve complex NLP problems, such as document summarization. Several neural network abstractive and extractive based models have been proposed [20]. For extractive summarization, a query-focused summarization approach was proposed by Cao et al. [21] to learn the representation of sentences and document cluster. They used the attention mechanism to learn the query relevance ranking and the saliency ranking simultaneously.

Furthermore, Yousefi-Azar et al. [22] introduced an unsupervised extractive query-oriented summarization model. They used autoencoder to get a concept vector for a sentence from the term-frequency (tf) input. Small random noise has been added to tf and used as input to the AE. The noisy AE model ranks the sentences and selects the top-ranked sentences to form the summary. Another extractive model proposed by Isonuma et al. 2017, [23]. In which, they used multitask-learning to address document summarization using curriculum learning for sentence extraction and document classification. Their framework has two components: one used for sentence extraction and the other for document classification. The document classification is used to learn common feature representation of salient sentences for summarization. The learning process is done using a Recurrent Neural Network (RNN) encoder-decoder architecture for sentence extraction and document classification. The summarization framework is quite complicated where it has four sub-modules: sentence encoder, document encoder, sentence extraction, and document classification. Recently, an extractive RNN-based summarization model proposed by Nallapati et al. [24], in which, the summarization task was treated as a classification problem where each sentence was visited sequentially and a binary decision is made to classify it as a summary or non-summary one.

In the context of abstractive summarization, several approaches have been proposed. The first work was carried out by Rush et al. [25]. It is an encoder-decoder model that used a convolutional network (CNN) and a feedforward neural network language model as encoder and decoder respectively. The attention mechanism is used to enhance the encoder. One of the shortcomings of this model is the use of CNN, which needs a fixed number of features. Moreover, only the first sentence of each article was used to generate the headline. Another recent abstractive work introduced by See et al. [26]. In which, they follow the encoder-decoder architecture where both the encoder and the decoder are RNNs. In addition, they introduce the pointer-generator network and the coverage mechanism to handle the out-of-vocabulary (OOV) problem and the repetition in the output respectively.

After exploring some deep learning-based model, it is worth mentioning two things: first, most of the previous deep learning-based models are supervised approaches, which need a huge amount of labeled data. Creating a suitable labeled data for text summarization is very challenging since the summarization task is subjective and in some cases, after a while, humans do not agree with their own judgment [7, 27]. In our work, we utilize word embeddings along with some traditional methods to build a robust unsupervised model. Second, most of the recent work has focused on headline generation tasks that means reducing one or two sentences to a single headline [26].

3. The proposed model

To get the summary, we proposed an LSA-based algorithm. In which, we used a word embedding based method instead of traditional methods to calculate the values of the input matrix (***A***). The new method is based on modified versions of the augment weight as a local weight and the entropy frequency as a global weight. The enhanced weighting schemes help the LSA algorithm to generate summaries with better quality. As shown in **Fig.** 1, the proposed model consists of two main stages: finding word vector and LSA algorithm implementation. The subsections 3.1 and 3.2 explain these stages in more details.

As a motivation for this work, we provide an overview of the limitations of the traditional weighting schemes in finding a good document representation. For example, according to traditional vector representation and after removing stop-words, the following two sentences: "Obama speaks to the media in Illinois" and "The President greets the press in Chicago" will have a zero similarity, although they have a similar meaning [18]. Moreover, term frequency (TF) is an essential part in almost all the traditional methods. Its value is calculated as the number of occurrences of a specific term in a sentence or a document. Depending on the exact matching, TF mostly fails to give a perfect representation since the writer usually uses different words to describe the same thing. **Table 1** shows the term frequency for all the terms in the two sentences in the previous example, excluding stop words. Although the two sentences are very similar, TF could not capture the semantic similarity between them. It is worth mentioning that we could use an external lexical database to overcome this to some extent, but this has some serious issues such as hard accurate word similarity, missing new words, and needs a lot of human feature engineering.

Table 1. Term frequency for s_1 and s_2 in the above example calculated as the number of occurrences of each term in the sentences

	Obama	speaks	media	Illinois	president	greets	press	Chicago
s_1	1	1	1	1	0	0	0	0
s_2	0	0	0	0	1	1	1	1

To solve this issue, we replace the term frequency, TF, with a new formula that able to work on the semantic level, Equation (7). The basic idea behind the new formula is replacing the term frequency with the cosine similarity between a word vector and the ones of every word in the sentence. **Table 2** shows the similarity between terms and sentences in the previous example calculated using the proposed formula, Equation (7). From the results in **Table 2**, it is notable that the new formula captures more information and gives a better representation so that it can be used as a replacement of the traditional term frequency in many weighting schemes. In this work, we use the new formula to improve the augment weight and the entropy frequency as discussed in Section 3.2.

Table 2. The term-sentence similarity matrix for sentences in the above example calculated using Equation (7)

	Obama	speaks	media	Illinois	president	greets	press	Chicago
s_1	1.603	1.269	1.283	1.548	0.328	0.749	1.013	1.279
s_2	0.773	1.028	0.860	0.708	1.239	1.346	1.363	1.109

3.1. Finding word vector

Learning word embedding is entirely unsupervised. Word2vec uses a simple neural network language model to learn a vector representation for each word using one of two ways, continuous bag of words or skip-gram. The network architecture of the latter composite of a projection layer between the input layer and the output one. It uses this simple architecture to predict nearby words. The model is trained on a very large unlabeled corpus to maximize the log probability of neighboring words [18].

Definition 1. Let *A* and *B* be two embedding distributions, if the similarity between *A* and *B* is high, then each embedding in *A* should be near to some embedding in *B*.

According to Definition 1, we can conclude that discovering such a complex relationship between the distributed representation of words compensates some of the key weaknesses of bag-of-words models [28]. In this work, we will use two pre-trained word embeddings, Google word2vec and GloVe, which are freely available. It is worth mentioning that we ignored the words that are not in the pretrained word embeddings from the representation.

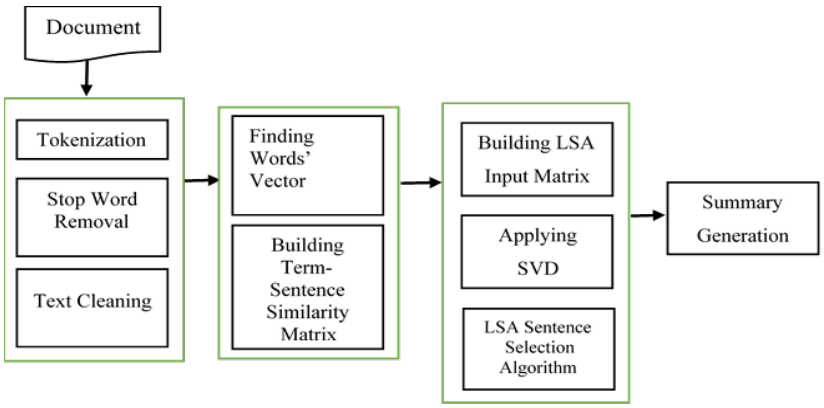


Fig. 1. The Proposed Model Architecture

3.1.1. Google word2vec Model[1]

Pre-trained Google News corpus (GoogleNewsvectors-negative300.bin.gz) with (about 100 billion words) includes an embedding for 3 million words/phrases with 300-dimension English word vectors, trained using the model in [15].

3.1.2. GloVe Model[2]:

GloVe [14] is another word embedding model. It learns by constructing co-occurrence statistics from a corpus (words X context), so instead of extracting the distributed representation from a neural network, as in word2vec, it optimizes the embeddings such that the product of two word vectors equals the log of the co-occurrence of those words within a predefined window size. In this work, we use the one trained on Common Crawl, (glove.840B.300d.zip), with 840B tokens, 2.2M vocab, cased, 300d vectors, and 2.03 GB file size.

3.2. LSA algorithm

LSA-based summarization models include three main steps: building the sentence - term matrix, LSA input matrix, SVD, and sentence selection step. In the following, we discuss each step from the perspective of our proposed model.

3.2.1. Building the input matrix

As the first step of LSA approach, the document should be represented as an $m \times n$ matrix, let it be A. The rows in this matrix represent words, while columns represent sentences. The cell value $a(w_i, s_j)$ represents the importance of the word w_i in the sentence s_j. When forming the input matrix, A, one can use a variety of weighting schemes that fall into two parts, a local weight based on the frequency within the sentence and a global one based on a word's frequency throughout the document [29]. In this work, as in Definition 2, the entry $a(w_i, s_j)$ is obtained by multiplying the local weight of the word w_i in the sentence s_j by the global weight of that word in the whole document.

Definition 2: Let D be the input document represented as an $m \times n$ matrix A, and let $A[j]$ be the jth column in that matrix, calculated using Equation (1), $a(w_i, s_j)$ is the weight for the ith word in the jth sentence, calculated using Equation (2):

$$A[j] = [\, a(w_1, s_j), a(w_2, s_j), \dots, a(w_i, s_j)] \quad (1)$$

$$a(w_i, s_j) = L(w_i, s_j) \times G(w_i) \quad (2)$$

, where: $L(w_i, s_j)$ is the Local Weight for the word w_i in the sentence s_j and $G(w_i)$ is the Global Weight for the word w_i in the whole document.

[1] https://code.google.com/archive/p/word2vec/
[2] http://nlp.stanford.edu/projects/glove/

There are different weighting schemes used to compute the local weight and the global weight for each word. The traditional ways for calculating these weights are difficult and inefficient because they must pass through a pre-processing step and they depend heavily on the exact matching. In this work, we propose a modified version of the augment weight as a local weight, Definition 3, and a modified version of the entropy frequency as a global weight, Definition 4. The following subsections describe two types of weighting schemes combinations, a traditional one and a word embedding-based one, as follows:

a) Traditional Weighting Scheme (AWEF)

The combination of the augment weight and the entropy frequency is a traditional way of building the input matrix. In this work, we used the following equations [30] to implement this weighting schemes used as a baseline:

(1) Augment Weight (AW): This weighting scheme is computed by Equation (3):

$$L(t_{ij}) = 0.5 + 0.5 \times (\frac{tf_{ij}}{tf_{\max}}) \quad (3)$$

Where tf_{ij} denotes the frequency of occurrence of the i^{th} word in the j^{th} sentence, and tf_{max} refers to the frequency of the most frequently occurring word in the j^{th} sentence.

(2) Entropy Frequency (*EF*): we used Equation (4) and Equation (5) to calculate (***EF***):

$$G(t_{ij}) = 1 + \sum \frac{P_{ij} \log_2 P_{ij}}{\log_2 n} \quad (4)$$

, where

$$P_{ij} = \frac{tf_{ij}}{gf_i} \quad (5)$$

Where n refers to the number of sentences, and gf_i is the number of occurrences of the i^{th} word in the entire document.

b) Embedding-Based Weighting Scheme (EMBAWEF)

Two embedding-based weighting schemes are proposed. The basic idea behind the new schemes is replacing the term frequency with the cosine similarity between the word vector and the ones of every word in the sentence.

(1) Embedding-Based Augment Weight (EMBAW): Definition 4 represents the embedding-based augment weight and **Algorithm 1** shows how it is calculated**.**

Definition 3. For an input document with m words and n sentences, let $D = (s_1, s_2, \dots, s_n)$ where s_j ($1 \le j \le n$) denotes the j^{th} sentence, W is a set of all terms in the document, and $V = (v_{w_1}, v_{w_2}, \dots, v_{w_m})$ where v_{w_i} $(1 \le i \le m)$ refers to the word vector of the term w_i. Let $L(w_i, s_j)$ be the local weight for term w_i in sentence s_j. For each w_i $(1 \le i \le m)$, the embedding-based augment weight for s_j is calculated by Equation (6):

$$L(w_i, s_j) = 0.5 + 0.5 \times (\frac{TermSentSim(w_i,s_j)}{TermSentSim_{\max j}}) \quad (6)$$

, where $TermSentSim(w_i, s_j)$ refers to the similarity score of the term w_i with sentence s_j, calculated using Equation (7), and $TermSentSim_{max j}$ refers to the similarity score of the term that has the maximum similarity with sentence s_j calculated using Equation (8). **Fig. 2** shows the way of calculating $TermSentSim$ for the term w_i with respect to sentence s_j.

$$TermSentSim(w_i, s_j) = \sum_{w' \in s_j} CosineSim(v_{w_i}, v_{w'}) \quad (7)$$

, where $CosineSim(v_{w_i}, v_{w'})$ denotes the cosine similarity between word vector of term w_i with respect to the ones of every term in sentence s_j.

$$TermSentSim_{max j} = max_{w \in W} TermSentSim(w, s_j) \quad (8)$$

(2) Embedding-Based Entropy Frequency (EMBEF): The embedding-based Entropy frequency is defined in Definition 5. **Algorithm 2** shows how it is calculated.

Definition 4. Let $G(w_i)$ be the global weight for w_i in D. For each $w_i (1 \leq i \leq m)$, the Embedding-Based Entropy Frequency for the word w_i is calculated by Equations (9) & (10):

$$G(w_i) = 1 + \sum_{j \in n} \frac{P(w_i,s_j) \log_2 P(w_i,s_j)}{\log_2 n} \quad (9)$$

, where n denotes the number of sentences in the document,

$$P(w_i, s_j) = \frac{TermSentSim(w_i,s_j)}{TermDocSim(w_i,D)} \quad (10)$$

, where $TermSentSim\ (w_i, s_j)$ refers to the similarity between word w_i and sentence s_j, calculated using Equation (7), and $TermDocSim(w_i, D)$ refers to the similarity score of word w_i with respect to the entire document calculated using Equation (11).

$$TermDocSim(w_i, D) = \sum_{w'' \in D} CosineSim(v_{w_i}, v_{w''}) \quad (11)$$

, where $CosineSim(v_{w_i}, v_{w''})$ denotes the cosine similarity between the word vector of word w_i with respect to the ones of every word in the entire document D.

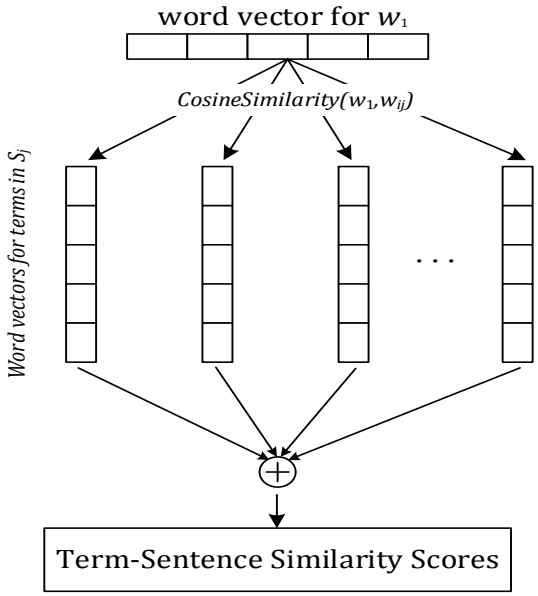


Fig. 2. Calculating Term-Sentence Similarity Scores

3.2.2. Singular Value Decomposition (SVD)

After getting $\boldsymbol{A}$ matrix using the new weighting schemes, we applied SVD on that matrix. SVD is a well-known algebraic algorithm used to identify the relationships between words and sentences [31] by breaking the input matrix into three matrices, as shown in Equation (12).

$$\boldsymbol{A} = \boldsymbol{U}\,\boldsymbol{\Sigma}\,\boldsymbol{V^T} \tag{12}$$

Where $\boldsymbol{U}$ is an $m \times n$ matrix that represents word by concept, $\boldsymbol{\Sigma}$ represents the scaling values. The importance of the concept can be determined by the singular values in $\boldsymbol{\Sigma}$ matrix. $\boldsymbol{V^T}$ represents concept-by-sentence. V^T represents concept-by-sentence, where its rows and columns represent concepts and sentences respectively. The most important concept can be determined by its position in the V^T rows where the first row represents the most important concept. The index of the highest value in a row determine the most related sentence to that concept. In this work, we used the SVD implementation in *scipy.sparse.linalg* python library to get the three matrices.

3.2.3. The algorithm of sentence selection

In this work, we implemented the popular algorithm proposed by [12] which proposes some improvements over some previous excellent work, as we mentioned in Section 2.1.

4. Complexity Analysis

In this section, the proposed approach algorithms are represented along with their complexity analysis. **Algorithm 1** is used to calculate the embedding-based augment weight (EMAW). The algorithm finds the word vector for a word and compares it with the vectors of every word in the document. Let $|W|$ be the number of words in the document, $|S|$ is the number of

sentences in the document and $|s_i|$ is the number of words in the longest sentence. The time complexity of this algorithm is determined by the most inner For loop, lines 5-8, that has the time complexity of $O(|W||S||s_i|)$; where $|S||s_i|$ is roughly equal $|W|$, the overall time complexity of **Algorithm 1** is $O(|W|^2)$. Comparing this algorithm with the traditional ones (AW), mentioned in section 3.2.1.a, will lead to the conclusion that they have the same complexity. The only difference is that the proposed algorithm needs to find the word vector from a lookup-table which takes $O(1)$ for each word, so the overall time complexity remains the same.

Embedding Augment Weight Algorithm

Input : a set of all terms in the document *W*, a set of sentences $S = (s_1, s_2, ..., s_j)$, a set of word vectors $V_w = (v_{w_1}, v_{w_2}, \dots, v_{w_m})$.

Output: Embedding Augment Weight (*EMBAW*)

1 **For** each word *w* in *W* **do**
2 $v_w = w2v(w)$
3 **For** each sentence s_i in *S* **do**
4 $TermSim := 0$
5 **For** each word *w'* in sentence s_j **do**
6 $v_{w'} = w2v(w')$
7 $TermSim := TermSim + (CosineSimilarity\ (v_w, v_{w'}))$
8 **End For**
9 $TermSentSim[w, s_j] := TermSim$
10 **End For**
11 **End For**
12 $TermSentSim_{max_j} := max(TermSentSim, axis = 0)$
13 **For** each word *w* in *W* **do**
14 **For** each sentence s_j in *S* **do**
15 $EMBAW[w, s_j] := (0.5 + 0.5 \times TermSentSim[w, s_j]/TermSentSim_{max_j}[s_j])$
16 **End For**
17 **End For**

Algorithm 1. Pseudocode for calculating Embedding-Based Augment weight (EMBAW).

Algorithm 2 is used to calculate the Embedding Entropy weight EMDEF. The term sentence similarity matrix (*TermSentSim*), calculated by **Algorithm 1**, is fed as input to **Algorithm 2**. The time complexity of this algorithm relies on the execution time of the most inner For loop, lines 4-8, that have the time complexity bounded to $O(|W||S|)$. This has the same complexity as the algorithm calculating the tradition Entropy Frequency (EF), mentioned in Section 3.2.1.a.

Embedding Entropy Frequency Algorithm

Input : a set of all words in the documents *W*, a set of sentences $S = (s_1, s_2, s_3, ..., s_j)$, and Term-Sentence Similarity Matrix $TermSentSim$

Output: Embedding Entropy Frequency (*EMBEF*)

$TermDocSim[w, D] := Sum(TermSentSim, axis = 1)$
For each word *w* in *W* **do**
For each sentence s_i in *S* **do**
$P[w] := TermSentSim[w, s_j] \ / \ TermDocSim[w, D]$
if $P[w] > 0$
$Plg[w].append(P[w] \times log_2 P[w])$

```
            else
              Plg[w].append(0)
            End if
          End For
0       End For
1       For each word w in W do
2         EMBEF[w]: = 1 + Sum(Plg[w]/ log_2 |S|)
3       End For
4
```

Algorithm 2. Pseudocode for calculating Embedding-Based Entropy Frequency (EMDEF).

5. The experiments and analysis

We compare the proposed approach, EMBAWEF, with the baseline models on three well-known datasets.

5.1. Datasets:

The proposed LSA based algorithm is evaluated on the three well-known datasets, DUC 2002, DUC 2004 and Multilingual 2015 Single-document Summarization (MSS 2015) [32]. DUC 2002 dataset includes 567 news articles categorized to 59 different clusters per topic. Alongside each document, there is a 100-word manual summary (single-document summarization) and for each cluster, there is a 100-word summary (multi-document summarization). We evaluate our model on the single-document summarization task. The second dataset, DUC 2004, includes five tasks. The first task, Task one, includes 500 news articles, each of which has four short gold standard summaries with a maximum length of 75 characters. Task two consists of fifty clusters of related documents with ten documents each. For each cluster, there are four human summaries with a maximum length of 665 characters (about 100 words) for each summary. The third dataset is Multilingual 2015 which contains 30 documents for each language out of 38 selected languages. We use the Single-document Summarization (MultiLing 2015) [3] task [32]. In which, each document is provided with one gold standard summary. We use ROUGE-1 and ROUGE-2[4] to evaluate our model on the English part of this dataset.

5.2. Baselines:

To evaluate the proposed approaches, we make an extensive comparison with multiple abstractive and extractive baselines on the three datasets as follows:

[3] http://multiling.iit.demokritos.gr/pages/view/1532/taskmss-single-document-summarization-data-and-information
[4] ROUGE-1.5.5 with options -n 2 -2 4 -u -x -m

5.2.1. On DUC-2002:

- Lead-3: this baseline simply chooses the first three sentences of the document as a summary.
- SummaRuNNer: RNN-based model by Nallapati et al. [24] , mentioned in Section 2.3.
- In addition, the well-known state-of-the-art (TF-IDF) is used as a baseline since it showed a competitive performance on DUC 2002.
- Cheng et al. [33] is an extractive model also used as a baseline on this dataset.

5.2.2. On DUC-2004:

- From the DUC-2004 single-document task, we include the PREFIX baseline that simply includes the first 75 characters of the document as a short summary.
- Neural attention-based model (ABS+) proposed by Rush et al. [25] is used as a baseline on DUC 2004, mentioned in section 2.3.
- We also report the TOPIARY system, which achieved the best performance in the DUC 2004 shared task.
- For the DUC-2004 multi-document task, we used LEAD that simply chooses the first 100 words from the most recent article in each cluster. Moreover, to ensure a fair comparison, we compare our model with three popular models applied on this dataset. The first is an RNN-based model proposed by Cao et al. [34] that used RNN for learning sentence embeddings. The second is a centroid-based method (C SKIP) for text summarization that exploits the compositional capabilities of word embeddings proposed by Rossiello et al. [16]. The third one is a neural graph-based model by Yasunaga et al.[35].

5.2.3. On Multiling MSS 2015:

- On this dataset, we used the BEST and The WORST scores obtained by the 23 participating systems. Moreover, we used the centroid-based model (C W2V) Rossiello et al. [16] as a baseline for English.

AWEF also serves as a baseline for the word embedding-based proposed model to measure the impact of using word embeddings while building the LSA input matrix. It is worth mentioning that during the implementation of the traditional TF-IDF and AWEF models, we employ the same LSA based selection algorithm with the same settings for selecting sentences for the summary.

5.3. Evaluation

The ROUGE metrics [36] are dominant for evaluating the summarization models. In which, the machine-generated summary is compared against one or several human-generated ones. In

this work, we used the pyrouge[5] python package and the ROUGE Toolkit[6] to evaluate the proposed models and the baselines.

Remark 2: we used ‘‘-l 100’’ and “-l b75” options in ROUGE toolkit command to truncate longer summaries in DUC 2002 and DUC 2004 respectively to ensure that the recall-only evaluation will be unbiased to length.

5.4. Experimental results

The proposed model, EMBAWEF, is compared with several baselines, mentioned in section 5.2. Furthermore, we implemented two LSA-based models as baselines for comparison. The first model is a combination of the traditional Augment weight and Entropy frequency (AWEF) as local and global weight respectively implemented and evaluated on the three datasets, DUC 2002, DUC 2004 and MSS 2015. The second one is the TF-IDF baseline implemented on the two datasets, DUC 2002 and DUC 2004, original documents. The SVD was applied on the matrix (A). After getting the three SVD matrices, the sentence selection algorithm of Steinberger et al. [12] was applied to select sentences for the summary. The generated summaries were compared against the reference ones in the three datasets. From the obtained experimental results in **Table 3**, **Table 4**, **Table 5, Table 6** and **Fig. 3**, we can make the following observations:

- The ROUGE scores in **Table 3** and **Fig. 3** show that the proposed model, EMBAWEF, performs the best on DUC 2002 dataset in terms of all ROUGE metrics used in this experiment. This asserts that using word embedding in the weighting scheme leads to an increase in the effectiveness of detecting semantically similar sentences, compared to traditional methods. Furthermore, an important implication of the obtained results is that word embedding models have reached a level where they can be utilized in a generic approach to delivering a feature representation that can be utilized to achieve state-of-the-art performance on an NLP task, such as document summarization.
- The model AWEF performed better than the baselines TF-IDF and LEAD-3. These results imply that combining AW and EF as a weighting scheme can capture more complex meanings than the other traditional combination of weighting schemes but it could not beat SummaRuNNer [24]. On the other hand, using TF-IDF as a weighting scheme to calculate the input matrix performed the worst for all ROUGE metrics used in this work, **Table 3**. A possible reason is that a wide variety of expressions by users made it difficult to calculate the semantic similarity. In the case of ROUGE-2, LEAD-3 performed better than AWEF. According to Lin et al. [37], the higher order ROUGE-N is worse than ROUGE-1 since it tends to score grammaticality rather than content.

[5] http://www.berouge.com/Pages/default.aspx

[6] ROUGE-1.5.5 with options: -n 2 -m -u -c 95 -r 1000 -f A -p 0.5 -t 0

Table 3. The proposed model performance using Word2Vec on DUC 2002

Model	Word2Vec on DUC 2002		
	ROUGE-1	ROUGE-2	ROUGE-L
EMBAWEF	**51.17%**	**23.43%**	**46.86%**
AWEF	45.13%	18.15%	41.06%
TF-IDF	43.57%	17.46%	39.15%
LEAD-3	43.60%	21.00%	40.20%
SummaRuNNer [24]	47.36%	22.10%	42.03%
Cheng et al '16 [33]	47.40%	23.00%	43.50%

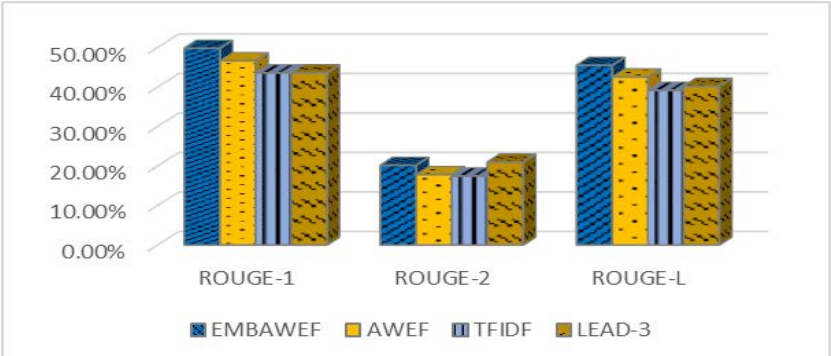


Fig. 3. The proposed model performance using GloVe on DUC 2002

- In the case of DUC 2004, **Table 4**, the results show that the proposed model, EMBAWEF, performed better than all the non-deep learning based baselines in terms of all ROUGE metrics used in this experiment, but it could not beat the deep learning model [25]. There are two possible reasons for this. The first one is that the ABS+ model has been trained on a huge dataset, annotated Gigaword dataset [38]. The model is headline-generation which turn to achieve a higher ROUGE score, but they usually fail when they are asked to generate a longer summary [39] . However, the proposed model achieves a competitive performance. The second one is that the attentional encoder-decoder models, such as Rush et al. [25], tend to produce short abstractive summaries with high ROUGE scores. However, the high ROUGE scores do not guarantee the readability and the correctness of the summaries since ROUGE metrics are a matter of calculating the n-gram overlap between the system summary and a reference one. One potential issue of the generative models is that they are optimizing for a ROUGE metric which leads to an increase in the scores compromising on the quality of the generated summary [39, 40]. This justifies the fact that our model could not beat the abstractive baseline, ABS+ [25], for short summaries, as in **Table 4**.

Table 4. The proposed model performance using using Word2Vec on DUC 2004 Single-Document Summarization

Model	Word2Vec on DUC 2004		
	ROUGE-1	ROUGE-2	ROUGE-L
Ours (EMBAWEF)	27.40%	7.41%	22.95%
AWEF	25.68%	6.39%	21.32%
TF-IDF	23.51%	4.03%	19.69%
PREFIX	21.43%	6.04%	17.45%

TOPIARY	25.12%	6.46%	20.12%
ABS+ [25]	28.18%	8.49%	23.81%
RAS-Elman [41]	**28.97%**	8.26%	24.06%
words-lvt2k-1sent [42]	28.35%	**9.46%**	**24.59%**

- To provide a fair comparison, we evaluated the model on the multi-document summarization task of DUC-2004 that has an average length of 100 words (665 characters) for each summary instead of 75 characters for the single-document summarization task. As shown in **Table 5**, the proposed model performs well and outperforms all the baselines, which asserts the feasibility of the proposed model to produce good summaries.

Table 5. The proposed model performance using Word2Vec on Multi-document Summarization Task of DUC-2004.

Model	Word2Vec on DUC 2004	
	ROUGE -1	ROUGE -2
Ours (EMBAWEF)	**40.23%**	**10.41%**
LEAD	32.42%	6.42%
C SKIP[16]	38.81%	9.97%
RNN[34]	38.78%	9.86%
GRU+GCN [35]	38.23%	9.48%

- The experimental results in **Table 6** show that the model (EMBAWEF) performs better than the baselines on MSS 2015 dataset. Also, it outperformed the centroid-based model proposed by Rossiello et al. [16] used as a baseline in this work.
- **Fig. 4** demonstrates that the model with word2vec has outperformed the one with GloVe.

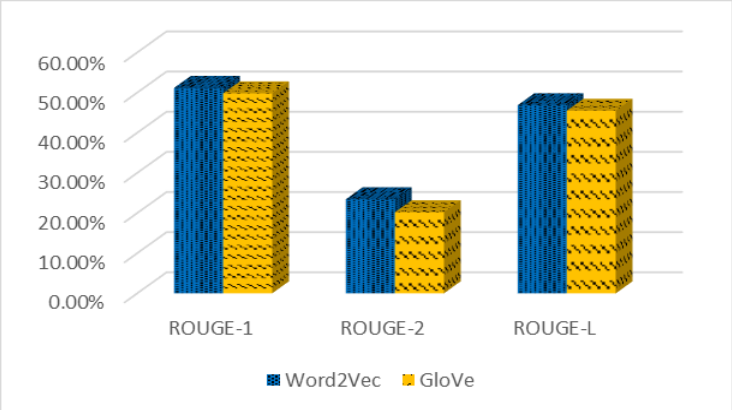


Fig. 4. The proposed model EMBAWEF performance using GloVe and Word2Vec on DUC 2002

- Finally, the EMBAWEF model achieved good ROUGE scores competing with the state-of-the-art models. Moreover, we consider an example of one of the extracted summaries from DUC 2002 using the model (EMBAWEF), shown in Appendix, **Fig. 5**.

Table 6. The proposed model performance for English on MultiLing2015

Model	English	
	ROUGE -1	ROUGE -2
Worst	37.17%	9.93%
Best	50.38%	14.12%
C W2V [16]	50.43%	13.34%
Ours (EMBAWEF)	**51.59%**	**15.41%**

6. Conclusion and future work

Traditional approaches to extract important information from a document rely heavily on human engineering features. In this paper, word embeddings are utilized to enhance the latent semantic analysis (LSA) input matrix weighting schemes. The proposed model, EMBAWEF, is used to compute the cell values for LSA input matrix. Applying Singular value decomposition algorithm on this matrix yields three matrices that are used to select sentences for the summary. The experimental results on the three datasets, DUC 2002, DUC 2004, and MultiLing 2015, show that the proposed models improve the performance of LSA algorithm in document summarization, especially EMBAWEF. The results also show the applicability of the model to extract the important sentences from the source effectively. The model achieves higher ROUGE scores than several well-known approaches. Although the new weighting schemes are evaluated on the document summarization task, it can be used in other information retrieval and NLP applications such as text similarity and web search. In the future work, we will try to apply the proposed weighting schemes to enhance the performance of other information retrieval applications.

7. ACKNOWLEDGEMENTS

We are grateful to the support of the National Natural Science Foundation of China (Grant No. 61379109, M1321007) and Science and Technology Plan of Hunan Province (Grant No. 2014GK2018 ,2016JC2011). We would like to thank the anonymous referees for their helpful comments and suggestions.

8. References

[1] Y.-H. Hu, Y.-L. Chen, and H.-L. Chou, "Opinion mining from online hotel reviews – A text summarization approach," *Information Processing & Management*, vol. 53, no. 2, pp. 436-449, 2017/03/01/ 2017. Article (CrossRef Link).

[2] S. Xiong and D. Ji, "Query-focused multi-document summarization using hypergraph-based ranking," *Information Processing & Management*, vol. 52, no. 4, pp. 670-681, 2016/07/01/ 2016. Article (CrossRef Link).

[3] J.-g. Yao, X. Wan, and J. Xiao, "Recent advances in document summarization," *Knowledge and Information Systems*, journal article March 28 2017. Article (CrossRef Link).

[4] Y. Sankarasubramaniam, K. Ramanathan, and S. Ghosh, "Text summarization using Wikipedia," *INFORMATION PROCESSING & MANAGEMENT*, vol. 50, no. 3, pp. 443-461, 2014. Article (CrossRef Link).

[5] D. Sarkar, "Text Summarization," in *Text Analytics with Python: A Practical Real-World Approach to Gaining Actionable Insights from your Data*Berkeley, CA: Apress, 2016, pp. 217-263. Article (CrossRef Link).

[6] E. Triantafillou, J. R. Kiros, R. Urtasun, and R. Zemel, "Towards generalizable sentence embeddings," in *Proc. of the 1st Workshop on Representation Learning for NLP*, Berlin, Germany, pp. 239–248, 2016. Article (CrossRef Link)..

[7] K. Al-Sabahi, Z. Zhang, J. Long, and K. Alwesabi, "An Enhanced Latent Semantic Analysis Approach for Arabic Document Summarization," *Arabian Journal for Science and Engineering*, journal article May 05 2018. Article (CrossRef Link).

[8] X. Han, T. Lv, Q. Jiang, X. Wang, and C. Wang, "Text summarization using Sentence-Level Semantic Graph Model," in *2016 4th International Conference on Cloud Computing and Intelligence Systems (CCIS)*, 2016, pp. 171-176: IEEE. Article (CrossRef Link).

[9] C. D. Boom, S. V. Canneyt, S. Bohez, T. Demeester, and B. Dhoedt, "Learning Semantic Similarity for Very Short Texts," in *2015 IEEE International Conference on Data Mining Workshop (ICDMW)*, 2015, pp. 1229-1234. Article (CrossRef Link).

[10] T. Kenter and M. d. Rijke, "Short Text Similarity with Word Embeddings," presented at the Proceedings of the 24th ACM International on Conference on Information and Knowledge Management, Melbourne, Australia, 2015. Article (CrossRef Link).

[11] Z. Wu *et al.*, "A topic modeling based approach to novel document automatic summarization," *Expert Systems with Applications*, vol. 84, pp. 12-23, 10/30/ 2017. Article (CrossRef Link).

[12] J. Steinberger and K. Jezek, "Using latent semantic analysis in text summarization and summary evaluation," in *Proc. ISIM'04*, 2004, pp. 93-100. Article (Google Scholar)

[13] Y. Shen *et al.*, "A Latent Semantic Model with Convolutional-Pooling Structure for Information Retrieval," in *Proceedings of the 23rd ACM International Conference on Conference on Information and Knowledge Management*, Shanghai, China, 2014, pp. 101-110, 2661935: ACM. Article (CrossRef Link)..

[14] J. Pennington, R. Socher, and C. D. Manning, "GloVe: Global Vectors for Word Representation," in *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, Doha, Qatar, 2014, pp. 1532-1543: Association for Computational Linguistics. Article (CrossRef Link)..

[15] T. Mikolov, K. Chen, G. Corrado, and J. Dean, "Efficient estimation of word representations in vector space," *CoRR*, vol. abs/1301.3781, 2013. http://arxiv.org/abs/1301.3781

[16] G. Rossiello, P. Basile, and G. Semeraro, "Centroid-based Text Summarization through Compositionality of Word Embeddings," in *Proceedings of the MultiLing 2017 Workshop on Summarization and Summary Evaluation Across Source Types and Genres*, Valencia, Spain, 2017, pp. 12-21: Association for Computational Linguistics. Article (CrossRef Link).

[17] H. Kobayashi, M. Noguchi, and T. Yatsuka, "Summarization Based on Embedding Distributions," in *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing*, Lisbon, Portugal, 2015, pp. 1984-1989: Association for Computational Linguistics. Article (CrossRef Link).

[18] M. Kusner, Y. Sun, N. Kolkin, and K. Weinberger, "From Word Embeddings To Document Distances," presented at the Proceedings of the 32Nd International Conference on International Conference on Machine Learning, Lille, France, 2015. Available: http://dl.acm.org/citation.cfm?id=3045118.3045221.

[19] J. Wieting, M. Bansal, K. Gimpel, and K. Livescu, "Towards universal paraphrastic sentence embeddings," *CoRR*, vol. abs/1511.08198, 2015. http://arxiv.org/abs/1511.08198

[20] K. Al-Sabahi, Z. Zhang, and M. Nadher, "A Hierarchical Structured Self-Attentive Model for Extractive Document Summarization (HSSAS)," *IEEE Access*, vol. 6, pp. 24205-24212, 2018. Article (CrossRef Link).

[21] Z. Cao, W. Li, S. Li, and F. Wei, "AttSum: Joint Learning of Focusing and Summarization with Neural Attention," in *Proceedings of COLING 2016, the 26th International Conference on Computational Linguistics: Technical Papers*, Osaka, Japan, 2016, vol. abs/1604.0, pp. 547--556: The COLING 2016 Organizing Committee. Article (CrossRef Link).

[22] M. Yousefi-Azar and L. Hamey, "Text summarization using unsupervised deep learning," *Expert Systems with Applications*, vol. 68, pp. 93-105, 2017/02/01/ 2017. Article (CrossRef Link).

[23] M. Isonuma, T. Fujino, J. Mori, Y. Matsuo, and I. Sakata, "Extractive Summarization Using Multi-Task Learning with Document Classification," in *Proc. of the 2017 Conference on Empirical Methods in Natural Language Processing*, pp. 2091-2100, 2017. Article (CrossRef Link).

[24] R. Nallapati, F. Zhai, and B. Zhou, "SummaRuNNer: A Recurrent Neural Network Based Sequence Model for Extractive Summarization of Documents," presented at the Proceedings of the Thirty-First AAAI Conference on Artificial Intelligence (AAAI-17), 2017. Available: http://aaai.org/ocs/index.php/AAAI/AAAI17/paper/view/14636.

[25] A. M. Rush, S. Chopra, and J. Weston, "A neural attention model for abstractive sentence summarization," *CoRR*, vol. abs/1509.00685, 2015. http://arxiv.org/abs/1509.00685

[26] A. See, P. J. Liu, and C. D. Manning, "Get To The Point: Summarization with Pointer-Generator Networks," in *Proc. of the 55th Annual Meeting of the Association for Computational Linguistics*, Vancouver, Canada, vol. 1, pp. 1073-1083: Association for Computational Linguistics, 2017. Article (CrossRef Link).

[27] M. Fuentes, E. González, D. FERRes, and H. RODRiguez, "QASUM-TALP at DUC 2005 Automatically Evaluated with a Pyramid based Metric," in *Proc. of Document Understanding Workshop (DUC). Vancouver, BC, Canada*, 2005. Article (Google Scholar)

[28] D. Kim and J. H. Lee, "Multi-document Summarization by Creating Synthetic Document Vector Based on Language Model," in *Proc. of 2016 Joint 8th International Conference on Soft Computing and Intelligent Systems (SCIS) and 17th International Symposium on Advanced Intelligent Systems (ISIS)*, pp. 605-609, 2016. Article (CrossRef Link).

[29] A. Kontostathis, "Essential Dimensions of Latent Semantic Indexing (LSI)," in *Proc. of System Sciences, 2007. HICSS 2007. 40th Annual Hawaii International Conference on*, pp. 73-73: IEEE, 2007. Article (CrossRef Link).

[30] C. D. Manning, P. Raghavan, and H. Schütze, *Introduction to Information Retrieval*. Cambridge University Press, p. 496, 2008.

[31] J.-H. Lee, S. Park, C.-M. Ahn, and D. Kim, "Automatic generic document summarization based on non-negative matrix factorization," *Information Processing & Management*, vol. 45, no. 1, pp. 20-34, 1// 2009. Article (CrossRef Link).

[32] G. Giannakopoulos *et al.*, "Multiling 2015: multilingual summarization of single and multi-documents, on-line fora, and call-center conversations," *Proceedings of SIGDIAL, Prague*, pp. 270-274, 2015. Article (CrossRef Link).

[33] J. Cheng and M. Lapata, "Neural Summarization by Extracting Sentences and Words," in *Proc. of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 484-494: Association for Computational Linguistics, 2016. Article (CrossRef Link).

[34] Z. Cao, F. Wei, L. Dong, S. Li, and M. Zhou, "Ranking with recursive neural networks and its application to multi-document summarization," presented at the Proceedings of the Twenty-Ninth AAAI Conference on Artificial Intelligence, Austin, Texas, 2015. Available: Article (Google Scholar).

[35] M. Yasunaga, R. Zhang, K. Meelu, A. Pareek, K. Srinivasan, and D. Radev, "Graph-based Neural Multi-Document Summarization," in *Proceedings of the 21st Conference on Computational Natural Language Learning (CoNLL 2017)*, Vancouver, Canada, pp. 452–462: Association for Computational Linguistics, 2017. Article (CrossRef Link).

[36] C.-Y. Lin, "Rouge: A package for automatic evaluation of summaries," in "Text summarization branches out: Proceedings of the ACL-04 workshop," Association for Computational Linguistics, Barcelona, SpainJuly, vol. 8, 2004. Available: http://aclweb.org/anthology/W04-1013.

[37] C.-Y. Lin and E. Hovy, "Automatic evaluation of summaries using N-gram co-occurrence statistics," in *Proc. of the 2003 Conference of the North American Chapter of the Association*

for Computational Linguistics on Human Language Technology - Volume 1, Edmonton, Canada, pp. 71-78, 1073465: Association for Computational Linguistics, 2003. Article (CrossRef Link).

[38] C. Napoles, M. Gormley, and B. V. Durme, "Annotated Gigaword," in *Proc. of the Joint Workshop on Automatic Knowledge Base Construction and Web-scale Knowledge Extraction*, Montreal, Canada, 2012. Article (Google Scholar).

[39] R. Paulus, C. Xiong, and R. Socher, "A Deep Reinforced Model for Abstractive Summarization," *CoRR*, vol. abs/1705.04304, 2017. http://arxiv.org/abs/1705.04304

[40] C.-W. Liu, R. Lowe, I. Serban, M. Noseworthy, L. Charlin, and J. Pineau, "How NOT To Evaluate Your Dialogue System: An Empirical Study of Unsupervised Evaluation Metrics for Dialogue Response Generation," in *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing*, Austin, Texas, pp. 2122-2132: Association for Computational Linguistics, 2016. Article (CrossRef Link).

[41] S. Chopra, M. Auli, and A. M. Rush, "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*, San Diego, California, pp. 93-98: Association for Computational Linguistics, 2016. Article (CrossRef Link).

[42] R. Nallapati, B. Zhou, C. Gulcehre, and B. Xiang, "Abstractive text summarization using sequence-to-sequence rnns and beyond," in *Proc. of the 20th SIGNLL Conference on Computational Natural Language Learning (CoNLL)*, Berlin, Germany, pp. 280-290: Association for Computational Linguistics, 2016. Article (CrossRef Link).

9. Appendix

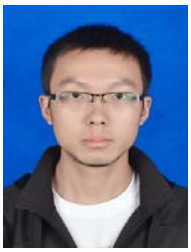
Fig. 6 is a snapshot from the implementation of this work in python 3.6. The example shows one representative document from DUC2002 (D081A/AP891103-0200) along with its gold and system summaries. It demonstrates that the model EMBAEF obtains a good performance identifying the key sentences in the document.

```
Python 3.6.0 |Anaconda 4.3.0 (64-bit)| (default, Dec 23 2016, 11:57:41) [MSC v.1900 64 bit (AMD64)]
Type "copyright", "credits" or "license" for more information.

          10:27:28,264 : INFO : loading projection weights from   /Summarization/data/GoogleNews-
vectors-negative300.bin
Reloaded modules: normalization, contractions, utilsfunctions, WV_Avg, LSAEmbeddingMatrixBuilder_V2
          10:29:33,627 : INFO : loaded (3000000, 300) matrix from   /Summarization/data/
GoogleNews-vectors-negative300.bin
Document Name: D081.P.AP891103-0200
Number of Sentences in The Document: 14

The Original Document:

{'D081.P.AP891103-0200': 'Thousands of miners in the northern Vorkuta region are expanding their
strike and some are blocking coal shipments the Soviet news media reported Friday. The miners are
demanding the government fulfill promises of improved living and job conditions. Soviet officials
have said the strikes could force fuel rationing during the Soviet Union s severe winter. Premier
Nikolai I. Ryzhkov introduced a bill in parliament that would increase pension benefits by about
percent and upgrade benefits for coal miners the official Tass news agency reported. The
Komsomolskaya Pravda youth newspaper in an article giving a sympathetic view of the workers
condition s in the Arctic Cirle coal mining districts said many miners do not see daylight for
months because of their underground work and the sun dips below the horizon in the winter. A
regional court in Vorkuta ruled that the latest round of strikes is illegal but did not impose any
penalties. The miners s unions said the decision will be appealed. Coal miners in the northern
region and the Ukraine struck for two weeks this summer but returned to work in July after
parliament passed a resolution promising reforms including improved social and economic conditions.
The miners say the government has renedged on its promises. The news media gave these accounts of
the latest strikes Workers at the Vorgashor mine in the Arctic Circle the largest mine in the
Vorkuta region continued their strike for an eighth day according to Tass. Komsomolskaya Pravda
said night shift workers walked out at three mines in Vorkuta and at another one the miners were
still working but were preventing the coal from being shipped outside the region. In the Ukraine s
Donetsk Coal Basin the nation s major coal producing area miner representatives were discussing
another strike. Tens of thousands of miners walked off the job for two hours Wednesday. Tass said
that in Donetsk the miners gathered in front of the House of Unions to demand that the government
set a deadline for implementing reforms. '}

The Gold Summary

Miners in the northern Vorkuta region are expanding their strike and blocking coal shipments,
demanding the government fulfill promises of improved living and job conditions.  In the Arctic
Circle coal-mining districts, miners do not see daylight for months because of their underground
work and the sun dips below the horizon in winter.
Coal miners in northern Ukraine struck for two weeks this summer but returned to work after a
resolution was passed promising reforms.  The miners say the government has reneged on its
promises.
A bill introduced by Premier Ryzhkov would increase pension benefits by 40% and upgrade
miners' benefits./n

The Generated Summary Using EMBAWEFLSA Model:

The Komsomolskaya Pravda youth newspaper in an article giving a sympathetic view of the workers
condition s in the Arctic Cirle coal mining districts said many miners do not see daylight for
months because of their underground work and the sun dips below the horizon in the winter.
Coal miners in the northern region and the Ukraine struck for two weeks this summer but returned to
work in July after parliament passed a resolution promising reforms including improved social and
economic conditions.
The news media gave these accounts of the latest strikes Workers at the Vorgashor mine in the
Arctic Circle the largest mine in the Vorkuta region continued their strike for an eighth day
according to Tass.
```

Fig. 5. Example document, gold summary and system summary from DUC 2002 using EMBAWEF Model and word2vec

Al-Sabahi Kamal received the B.S. degree in computer science from Sana'a University, Sana'a, Yemen, in 2008 and the M.S. degree in information technology from OUM University, Kuala Lumpur, Malaysia, in 2015. He is currently pursuing the Ph.D. degree in computer science at Central South University, School of Information Science and Engineering, Hunan, Changsha, China. His research interest includes deep learning, natural language processing, Knowledge Engineering and data mining.

Zhang Zuping received BS degree in Foundation of Mathematics, Hunan Normal University in 1989, received the MS degree in Foundation of Mathematics, Jilin University in 1992, received the Ph.D. degree in Computer Application Technology, Central South University in 2005. He is now a Professor in School of Information Science and Engineering, Central South University, Chang-Sha, China. His current research interests include information fusion and information system, bigdata technology and application, parameter computing and biology computing.



Yang Kang received B.S. degree in Computer Science and Technology, Wuhan Institute of Technology in 2016, He is now a Master Student in School of Information Science and Engineering, Central South University, Chang Sha, China. His current research interests include Machine Learning, Natural language Processing.