Multi-document Extractive Summarization Using Window-based Sentence Representation

Yong Zhang
School of Electrical and Electronic Engineering
Nanyang Technological University
Singapore
Email: yzhang067@e.ntu.edu.sg

Meng Joo Er
School of Electrical and Electronic Engineering
Nanyang Technological University
Singapore
Email: emjer@ntu.edu.sg

Rui Zhao
School of Electrical and Electronic Engineering
Nanyang Technological University
Singapore
Email: rzhao001@e.ntu.edu.sg

***Abstract*—Multi-document summarization has gained popularity in many real world applications because significant information can be obtained within a short time. Extractive summarization aims to generate a summary of a document or a set of documents by ranking sentences and the ranking results rely heavily on the quality of sentence features. However, almost all previous algorithms require hand-crafted features for sentence representation. In this paper, we leverage on word embedding to represent sentences so as to avoid the intensive labor of feature engineering. We propose a new technique, namely *window-based sentence representation (WSR)*, to obtain the features of sentences using pre-trained word vectors. The method is developed based on the Extreme Learning Machine (ELM). Our proposed framework does not require any prior knowledge and thus can be applied to various document summarization tasks with different languages, written styles and so on. We evaluate our proposed method on the DUC 2006 and 2007 datasets. This proposed method achieves superior performance compared with state-of-the-arts document summarization algorithms with significantly faster learning speed.**

I. Introduction

Automatic text summarization has been widely researched in recent years with the explosive growth of accessible information due to the rapid development of Internet and computing technology. It can mainly be divided into two categories, namely *abstractive summarization* and *extractive summarization*. The abstractive summarization methods involve sentence compression and reformulation which resemble human summarizers, but the linguistic processing procedure is so complicated that it is not easy to be implemented [1]–[3]. However, the extractive methods directly extract the most informative sentences in a document to form the final summary. Due to its simplicity, most works done in this area fall in the extraction-based category.

Extractive document summarization methods roughly fall into three main categories [4], [5]: 1) Methods based on sentence positions and article structure, 2) Unsupervised methods, 3) Supervised methods. For the first category, important sentences, such as those in introductory or concluding part, will be selected to fit into the summary. One famous method (LEAD) is proposed in [6] which simply extracts the leading sentences to summarize a document. However, such methods can only be applied to strictly structured documents like newswire articles. On the other hand, unsupervised methods rank sentences by salience scores which are estimated based on statistical and linguistic features and extract the top ones to constitute the summary. Maximal Marginal Relevance (MMR) [7] is a well-known unsupervised method which obtains a trade-off between relevance and redundancy of sentences. The MMR framework is cast as an integer linear programming (ILP) in [8] to find the global optimal solution. Graph-based ranking methods play a significant role in unsupervised document summarization in recent years, such as the LexRank [9] and the TextRank [10]. They first build a graph of sentence similarities and then calculate the importance of a sentence by inspecting its links to all other sentences in the graph recursively. Some other unsupervised extraction-based document summarization methods include the Latent semantic analysis (LSA) [11], the Markov random walk (MRW) [12] and the submodularity-based methods [13]. In contrast to unsupervised methods, supervised approaches utilize a set of training documents together with their corresponding hand-crafted summaries to train a binary classifier which is used to predict whether a sentence should be included in the summary or not. In [14], a probabilistic approach termed conditional random field (CRF) is proposed to rank full sentences while researchers in [15], [16] train their models based on n-gram regression. Recently, some researchers [17], [18] measure the salience of both sentences and n-grams. The author of [17] takes advantage of support vector regression (SVR) and the approach in [18] is developed based on the recursive neural networks (RNN).

Almost all the aforementioned methods depend on hand-crafted features to represent sentences which result in intensive human labor. In this paper, we leverage on the word embedding technique to represent sentences. Word embedding has drawn great attention in recent years because it can capture both semantic and syntactic information. The idea of word embedding has a very long history but has become popular since Bengio *et al.*'s works [19], [20] in which each word is represented by a vector and the concatenation of several previous word vectors is employed to predict the next word using a neural networks language

978-1-4799-7560-0/15 $31.00 © 2015 IEEE
DOI 10.1109/SSCI.2015.67

model. Since then, a lot of researchers have explored the salient features of the vector representation of words [21]–[25] using neural networks. These models can map the words into a vector space where words with similar semantics lie close to each other, i.e., *vector("American")* is closer to *vector("France")* than to *vector("Bread")*.

Research beyond word level to explore phrase-level and sentence-level representation [25]–[29] has also been carried out. Socher *et al.* [28] uses a parse tree to combine word vectors to represent a sentence. The method can capture the sentiment of a sentence effectively, but the training procedure is very complicated. A much simpler method is to summarize all the word vectors in a sentence to obtain the representation of the sentence. However, this method will lose the word order just like conventional bag-of-words method. In this paper, we propose a new technique termed *window-based sentence representation (WSR)* to represent sentences which can be implemented as easily as the summarization method while preserving word order information. We employ the Extreme Learning Machine (ELM) of [30] to train our model. The training speed is hundreds or thousands of times faster compared with the parse tree method and deep neural networks. We evaluate our method on two summarization benchmark datasets DUC2006 and DUC2007. The proposed method achieves significantly superior performance compared with state-of-the-arts document summarization algorithms within a much shorter time.

This paper is organized as follows: Section II gives a brief review of related works. In Section III, the proposed algorithm is described in details. The performance of the proposed method is compared with other learning algorithms in Section IV. Conclusions are drawn in Section V.

II. Overview of Related Works

The two most related methods, namely word embedding and extreme learning machine (ELM) are briefly discussed in this section.

A. Word Embedding

Word embedding is also called word vector or distributed word representation, which has a long history. The idea of distributed word representations was first proposed in [31], [32] and other early works including [33], [34]. Recently, neural networks language models have been proposed for predicting probability distribution over the next word given some preceding words [19], [20]. In their framework, a feedforward neural network with a linear projection layer and a nonlinear hidden layer was used to learn jointly the word vector representation and a statistical language model. Another popular architecture of neural networks language model was proposed by Milkov *et al.* [24], [25]. They developed their methods in the context of recurrent neural networks and proposed two efficient word representation estimation models, namely Continuous Bag-of-Words (CBOW) model and Skip-gram model, both being among the most widely used distributed word representation methods nowadays.

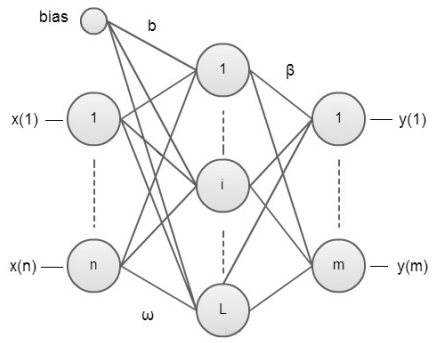

Fig. 1. ELM network structure.

Word embedding has been applied to many natural language processing applications such as name entity recognition [21], word sense disambiguation [23], parsing [28], tagging [35] and machine translation [36] and it has been demonstrated that they are very efficient in capturing semantic and syntactic information within texts. Going beyond word-level representation, a lot of efforts have been devoted to exploring how to represent phrases and sentences using vectors [25]–[29]. Socher *et al.* [37]–[40] mainly leverage on recursive neural networks with a tree structure to combine word vectors and their works are shown to work well for sentence-level representation. However, their model can hardly be extended beyond single sentences. Most recently, Le *et al.* [41] developed a method termed *Paragraph Vector* which can not only represent phrases and sentences but also be extended to the document level.

B. ELM

The ELM and its variants have been widely used due to their salient features such as tuning free and extreme fast learning speed. The relationship and differences between ELM and other related works can be found in a recent published work [42]. The ELM was first proposed by Huang *et al*. [30] for a single-hidden-layer feedforward neural network (SLFN) with randomly chosen input weights and hidden nodes and analytically determined output weights. The structure of the original ELM is shown in Fig. 1.

Let us suppose that there are K observations $(\mathbf{x}_k, \mathbf{t}_k)_{k=1}^{K}$ with $\mathbf{x} \in \mathbb{R}^{1\times n}$ and $\mathbf{t} \in \mathbb{R}^{1\times m}$. If a SLFN with L hidden nodes can approximate the data with zero error, the following equation holds:

$$\sum_{l=1}^{L} \boldsymbol{\beta}_l G(\mathbf{x}_k; \mathbf{a}_l, b_l) = \mathbf{t}_k, \quad k = 1, \ldots, K \tag{1}$$

where $\boldsymbol{\beta}_l$ are the output weights and $G(\mathbf{x}; \mathbf{a}, b)$ is the activation function where $\mathbf{a}, b$ are the parameters of hidden nodes. The term $G(\mathbf{x}_k; \mathbf{a}_l, b_l)$ represents the output of the lth hidden node with respect to the kth input data. The activation function can be both additive function and RBF function. For the additive function, the term $\mathbf{a}_l$ is the weight vector connecting the input layer to the lth hidden node and the term b_l is the bias of the lth hidden node. For the RBF function, the terms $\mathbf{a}_l$ and b_l are the center and impact

factor of the lth hidden node respectively. Equation (1) can be written in the following compact form:

$$\mathbf{H}\boldsymbol{\beta} = \mathbf{T} \tag{2}$$

where

$$\mathbf{H} = \begin{bmatrix} G(\mathbf{x_1}, \mathbf{a_1}, b_1) & \cdots & G(\mathbf{x}_1, \mathbf{a}_L, b_L) \\ \vdots & \vdots & \vdots \\ G(\mathbf{x}_K, \mathbf{a_1}, b_1) & \cdots & G(\mathbf{x}_K, \mathbf{a}_L, b_L) \end{bmatrix}$$

$$\boldsymbol{\beta} = \begin{bmatrix} \beta_1 & \cdots & \beta_L \end{bmatrix}^T \quad and \quad \mathbf{T} = \begin{bmatrix} \mathbf{t}_1 & \cdots & \mathbf{t}_K \end{bmatrix}^T$$

The term $\mathbf{H}$ is called the hidden layer output matrix of the network, $\mathbf{T}$ is the target matrix and $\boldsymbol{\beta}$ is the output weight matrix. The parameters of the hidden nodes $(\mathbf{a}_l, b_l)$ are randomly generated, and the only parameter needed to be calculated is the output matrix $\boldsymbol{\beta}$, which can be analytically determined using the least squares estimate (LSE) as follows:

$$\boldsymbol{\beta} = \mathbf{H}^{\dagger}\mathbf{T} \tag{3}$$

where $\mathbf{H}^{\dagger}$ is the Moore-Penrose generalized inverse of the matrix $\mathbf{H}$ which can be calculated through the orthogonal projection method, orthogonalization method and singular value decomposition (SVD) [43].

It should be highlighted that the number of hidden neurons is the only parameter to be specified in the ELM as opposed to other learning algorithms which usually have a lot of parameters to be tuned [44]. Another remark of the ELM is that the activation function for additive nodes can be any bounded non-constant piece-wise continuous function and the activation function for RBF nodes can be any integral piece-wise continuous function. Since the original ELM, many variants have been proposed and applied to all kinds of applications. The recent developments and future trend of ELM algorithm can be found in [45]

III. Window-based Sentence Representation for Multi-document Summarization

Extractive summarization is defined as the selection process of salient sentences in a corpus of documents to generate a brief summary which can best describe the subject. We denote the document corpus as $C = \{D_1, D_2, ...\}$, in which D_i is the ith document in the corpus. Each document consists of a set of sentences. We include all the sentences in the corpus to form the **candidate set** $CS = \{\mathbf{s_1}, \mathbf{s_2}, ..., \mathbf{s_n}\}$, where $\mathbf{s_i} \in \mathbb{R}^d$ is the vector representation of the ith sentence in the corpus. The term n is the number of distinct sentences in the corpus while d is the dimension of the sentence vector. The determination of the sentence representation has never been a trivial task. Most conventional methods require hand-crafted features which result in intensive human labor. Our proposed method leverages on pre-trained the word embedding technique to obtain sentence representations. Next, we employ the ELM to assign salience scores to sentences in the **candidate set**. After assignment of salience scores, the sentences can be ranked so that the sentences in the **candidate set** with high scores will be selected as summary sentences. The selected sentences form the **summary set** $SS = \{\mathbf{s_1^*}, \mathbf{s_2^*}, ..., \mathbf{s_k^*}\}$. Note that $k << n$ and $SS \subset CS$.

A. Preprocessing

Our proposed method is a supervised method. But ready-made salience scores are not available for training data. Therefore, we first pre-process the documents to obtain a salience score for each sentence. The document sets are given together with their reference summaries. We adopt the widely-accepted automatic summarization evaluation metric, ROUGE, to measure the salience. As ROUGE-1 (R_1) and ROUGE-2 (R_2) most agree with human judgment among all the ROUGE scores [46], we calculate each sentence's score as follows:

$$score = \alpha(R_1 + R_2) \tag{4}$$

We set the coefficient $\alpha = 0.5$ for equal weighting of the two scores. The terms R_1 and R_2 are obtained by comparing each sentence in multi-documents with corresponding reference summaries.

The other problem is to determine input features of the ELM model, i.e., obtain sentence representations. This is the focal point of our proposed method. With the aid of pre-trained word vectors, we can denote a sentence as $\mathbf{s_i} = \{\mathbf{w_1}, \mathbf{w_2}, ..., \mathbf{w_l}\}$, in which $\mathbf{w_i}$ is the word embedding of the ith word in the sentence and l is the length of the sentence. An intuitive and simple method to represent the sentence is to summarize all the word vectors as follows:

$$\mathbf{s} = \sum_{i=1}^{l} \mathbf{w_i} \tag{5}$$

However, this method loses the word order information in the sentence. In order to address the problem, we propose a *window-based sentence representation (WSR)* method which is able to preserve the order information. We represent a word taking into its context words instead of simply using its own word vector. The new word representation is denoted as the concatenation of word vectors within a context window. Supposing the window size is m, the new word representation, $\mathbf{v}_i$, is given by

$$\mathbf{v}_i = [\mathbf{w}^T_{i-\frac{m-1}{2}}, ..., \mathbf{w}^T_i, ..., \mathbf{w}^T_{i+\frac{m-1}{2}}]^T \in \mathbb{R}^{md} \tag{6}$$

where the term m should be assigned an odd number. For those words at the first or last positions, zero vectors are padded to guarantee the context window size. With the new representations of words, we can obtain the representation of a sentence as follows:

$$\mathbf{s} = \sum_{i=1}^{l} \mathbf{v}_i \tag{7}$$

This method is still very simple and easy to implement, but it is able to preserve some order information compared with the summarization method. In order to demonstrate the effectiveness of the context window, we compare our method with the summarization method in the experiment section. The summarization method is termed *summation sentence representation (SSR)* in this paper.

B. Training

After the preprocessing procedure, we use the training documents to train the ELM model. We use RBF kernel in the ELM model, and the centers and impact factors of the kernel are all randomly generated. The only parameter needed to be tuned is the number of hidden neurons. It has been proved that the single-hidden-layer neural network is a universal function approximator as long as the number of hidden neurons is big enough. As the input feature number is md, which is not a very large number, the number of hidden layer nodes can be selected slightly larger than md. The detail training procedure can be found in Section II-B. The objective function of the model is formulated as follows [47]:

$$min \sum_{i=1}^{n} \|score(\mathbf{s}_i) - \mathbf{h}(\mathbf{s}_i)\boldsymbol{\beta}\|_2^2 + \lambda \|\boldsymbol{\beta}\|_2^2 \qquad (8)$$

where $score(\mathbf{s}_i)$ is the true score of the ith sentence obtained in the preprocesing process, $\mathbf{h}(\mathbf{s}_i)$ is the corresponding hidden-layer output and $\boldsymbol{\beta}$ is the output weight matrix. The term λ is a user defined parameter constraining the value of $\boldsymbol{\beta}$. Equation (8) can also be written in a compact form:

$$min \|\mathbf{Y} - \mathbf{H}\boldsymbol{\beta}\|_2^2 + \lambda \|\boldsymbol{\beta}\|_2^2 \qquad (9)$$

where $\mathbf{H} = [\mathbf{h}(\mathbf{s}_1), \cdots, \mathbf{h}(\mathbf{s}_n)]^T$ is the hidden layer output matrix for all the sentences while $\mathbf{Y} = [score(\mathbf{s}_1), \cdots, score(\mathbf{s}_n)]^T$ is the output score vector.

This equation minimizes both training error and norm of output weights because both small training error and small norm of weights contribute to the improvement of generalization performance according to Bartlett's theory [48]. The output matrix $\boldsymbol{\beta}$ can be analytically determined using the following equation [47]:

$$\boldsymbol{\beta} = (\mathbf{H}^T\mathbf{H} + \lambda\mathbf{I})^{-1}\mathbf{H}^T\mathbf{Y} \qquad (10)$$

As there are few parameters to tune, the training process is very fast. This is a strong advantage over other conventional machine learning models.

C. Testing

For the testing documents, features of all the sentences are calculated just like in the training documents. Next, they will be used as the input to the trained ELM model to obtain their salience scores. Afterwards, sentences in each multi-document cluster can be ranked according to their salience scores. Since a good summary should be not only informative but also non-redundant, we employ a sentence selection method in [49] to select from the ranked sentences. The selection method queries the sentence with the highest salience score and adds it to the summary if the similarity of the sentence with all the sentences already existing in the summary does not exceed a threshold. This sentence selection procedure is especially necessary for multi-document summarization because sentences extracted from different documents in one topic can be very similar. The selection process repeats until the length limit of the final summary is met.

IV. Performance Evaluation

A. Datasets

The benchmark datasets from the Document Understanding Conferences (DUC[1]) are used to evaluate our proposed document summarization method. The datasets are English news articles. In this paper, we evaluate the performance of multi-document summarization on DUC 2006 and 2007 datasets. When evaluating on the DUC 2006 dataset, we train our model on DUC 2004 and 2007 datasets. Simultaneously, DUC 2004 and 2006 datasets are used to train the model for evaluation on the DUC 2007 dataset. The characteristics of the three-year datasets are given in Table I. The table also contains the length limit of automatic summary for each dataset. The reference summaries for each set of documents are given together with the testing documents.

B. Evaluation Metrics

For the evaluation of summarization performance, we employ the widely used ROUGE[2] toolkit [50]. ROUGE has become the standard evaluation metric for DUC since 2004. Rouge assesses the quality of an automatic summary by counting the overlapping units such as n-gram, common word pairs and longest common sub-sequences between the automatic summary and a set of reference summaries. The n-gram ROUGE metric (ROUGE-N) can be computed as follows:

$$ROUGE - N = \frac{\sum_{S \in \{RefSum\}} \sum_{n-gram \in S} Count_{match}(n-gram)}{\sum_{S \in \{RefSum\}} \sum_{n-gram \in S} Count(n-gram)} \qquad (11)$$

where n stands for the length of the n-gram, $Count(n-gram)$ is the number of n-grams in the set of reference summaries and $Count_{match}(n-gram)$ refers to the number of n-grams co-occurring in a system generated summary and the set of reference summaries. Among all the ROUGE scores, ROUGE-1 (unigram) and ROUGE-2 (bigram) have been demonstrated to most agree with human judgment [46]. Rouge-SU4 is also a very popular metric because it conveys the readability of a candidate summary. Therefore, ROUGE-1, ROUGE-2 and Rouge-SU4 scores are reported in our evaluation study. We set the length parameter "-l 250".

[1] http://duc.nist.gov

[2] ROUGE-1.5.5 with options: -n 2 -x -m -2 4 -u -c 95 -r 1000 -f A -p 0.5 -t 0 -d

TABLE I. CHARACTERISTICS OF DATA SETS

Year	Clusters	Documents/Cluster	Length Limit
2004	50	10	250 words
2006	50	25	250 words
2007	45	25	250 words

Rouge generates precision, recall and F-measure metrics. We employ F-measure, which is a combination of precision and recall metrics, in our experiment to compare the performance of different algorithms.

C. Parameter Settings

The embeddings of words used in our experiments are initialized using Skip-gram neural networks method proposed by Mikolve *et al.* [24] which is available in *word2vec* tool[3]. The model is trained on a part of Google News dataset which has about 100 billion words. The dimension of each word vector is 300. The word vectors are already available and can be directly downloaded from the *word2vec* tool website. Therefore, it is not necessary to spend a great amount of time to train and obtain word vectors. For the setting of ELM, the only parameter needed to be determined is the number of hidden neurons, which we set as 1000 in this paper. The python package on the ELM website[4] is employed for our experiment. The threshold value of sentence selection for multi-document summarization is set as 0.4 and the context window size is set as 3. These parameters are determined using cross-validation. The experiments are done on a 2.0GHz PC computer.

D. Comparative Studies

We compare our proposed method with several state-of-the-arts document summarization techniques which are briefly described as follows:

Random: Selects sentences randomly from the **candidate set**.

Lead [6]: First sorts all the documents chronologically and then extracts the lead sentence from each document to form the summary.

LSA [11]: Applies the singular value decomposition (SVD) on the term frequency matrix and then selects sentences with highest eigenvalues.

DSDR [51]: Uses sparse coding to represent each sentence as a nonnegative linear combination of the summary sentences.

Table II and Table III are the overall performance comparison results of our proposed method *WSR* against other state-of-the-arts algorithms. It is obvious that the proposed method *WSR* outperforms all the other algorithms significantly. Among all the algorithms, LSA gives the poorest performance on both datasets. LSA exploits SVD on term-frequency matrix and supposes that sentences with highest eigenvalues are most informative sentences in a document corpus. The experiment results suggest that such hypothesis may not be true for human understanding. The table also

[3]https://code.google.com/p/word2vec

[4]http://www.ntu.edu.sg/home/egbhuang/

TABLE II. SYSTEM COMPARISON RESULTS ON DUC2006

System	ROUGE-1	ROUGE-2	ROUGE-SU4
Random	0.28047	0.04613	0.08785
Lead	0.30758	0.04836	0.08652
LSA	0.24415	0.03022	0.07099
DSDR	0.32034	0.04585	0.09804
SSR	0.32804	0.04151	0.09725
WSR	**0.34416**	**0.05178**	**0.10954**

TABLE III. SYSTEM COMPARISON RESULTS ON DUC2007

System	ROUGE-1	ROUGE-2	ROUGE-SU4
Random	0.30199	0.04628	0.08763
Lead	0.31188	0.0576	0.10201
LSA	0.25977	0.04062	0.08338
DSDR	0.32641	0.04876	0.10245
SSR	0.35571	0.05623	0.1158
WSR	**0.37394**	**0.06961**	**0.12776**

shows that Lead performs a little better than the Random method, which may be because that article writers like to put summary sentences at the beginning of the documents. DSDR achieves the best performance on ROUGE-1 score except the two word embedding-based methods. It proves that sparse coding helps to find informative sentences out of documents. The two word-embedding-based methods, SSR and WSR, show the best performance, proving that word embedding is effective in capturing semantic and syntactic information. Comparing WSR with SSR, we can conclude that the incorporation of a context window indeed helps document summarization. This is because WSR preserves word order information compared with SSR.

In addition, our method is very efficient with a very fast learning speed. The average running time of DSDR on one document set is more than $4000s$ while the time of WSR is only about $8s$. Our algorithm is more than 500 times faster than DSDR. This results from the ELM's salient tuning-free features. Besides, the biggest advantage of our method is that it does not need hand-crafted features. The pre-trained word embedding has saved us an enormous amount of time and effort, enabling us to avoid the intensive human labor of feature engineering.

V. CONCLUSIONS

In this paper, a new technique termed *window-based sentence representation (WSR)* has been successfully developed to obtain the features of sentences based on pre-trained word vectors. The use of word embedding enables us to avoid the intensive human labor of feature engineering. We employ a context window to preserve word order information in sentences. The model is realized by the ELM. Our proposed framework does not require any prior knowledge and thus can be applied to various document summarization tasks with different languages, written styles and so on. We evaluate our proposed method on the DUC 2006 and 2007 datasets. This proposed method achieves superior performance compared with state-of-the-arts document summarization algorithms with significantly faster learning speed.

ACKNOWLEDGMENT

The authors would like to acknowledge the funding support from the Ministry of Education, Singapore (Tier 1 AcRF, RG30/14).

REFERENCES

[1] H. Daumé III and D. Marcu, "A noisy-channel model for document compression," in *Proceedings of the 40th Annual Meeting on Association for Computational Linguistics*. Association for Computational Linguistics, 2002, pp. 449–456.

[2] J. Turner and E. Charniak, "Supervised and unsupervised learning for sentence compression," in *Proceedings of the 43rd Annual Meeting on Association for Computational Linguistics*. Association for Computational Linguistics, 2005, pp. 290–297.

[3] A. F. Martins and N. A. Smith, "Summarization with a joint model for sentence extraction and compression," in *Proceedings of the Workshop on Integer Linear Programming for Natural Langauge Processing*. Association for Computational Linguistics, 2009, pp. 1–9.

[4] I. Mani and M. T. Maybury, *Advances in automatic text summarization*. MIT Press, 1999, vol. 293.

[5] K.-Y. Chen, S.-H. Liu, B. Chen, H.-M. Wang, W.-L. Hsu, and H.-H. Chen, "A recurrent neural network language modeling framework for extractive speech summarization," in *Multimedia and Expo (ICME), 2014 IEEE International Conference on*. IEEE, 2014, pp. 1–6.

[6] M. Wasson, "Using leading text for news summaries: Evaluation results and implications for commercial summarization applications," in *Proceedings of the 36th Annual Meeting of the Association for Computational Linguistics and 17th International Conference on Computational Linguistics-Volume 2*. Association for Computational Linguistics, 1998, pp. 1364–1368.

[7] J. Carbonell and J. Goldstein, "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*. ACM, 1998, pp. 335–336.

[8] R. McDonald, *A study of global inference algorithms in multi-document summarization*. Springer, 2007.

[9] G. Erkan and D. R. Radev, "Lexrank: graph-based lexical centrality as salience in text summarization," *Journal of Artificial Intelligence Research*, pp. 457–479, 2004.

[10] R. Mihalcea and P. Tarau, "Textrank: Bringing order into texts." Association for Computational Linguistics, 2004.

[11] Y. Gong and X. Liu, "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*. ACM, 2001, pp. 19–25.

[12] X. Wan and J. Yang, "Multi-document summarization using cluster-based link analysis," in *Proceedings of the 31st annual international ACM SIGIR conference on Research and development in information retrieval*. ACM, 2008, pp. 299–306.

[13] H. Lin and J. Bilmes, "A class of submodular functions for document summarization," in *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies-Volume 1*. Association for Computational Linguistics, 2011, pp. 510–520.

[14] M. Galley, "A skip-chain conditional random field for ranking meeting utterances by importance," in *Proceedings of the 2006 Conference on Empirical Methods in Natural Language Processing*. Association for Computational Linguistics, 2006, pp. 364–372.

[15] C. Li, X. Qian, and Y. Liu, "Using supervised bigram-based ilp for extractive summarization." in *ACL (1)*. Citeseer, 2013, pp. 1004–1013.

[16] K. Hong and A. Nenkova, "Improving the estimation of word importance for news multi-document summarization," in *Proceedings of EACL*, 2014.

[17] Y. Hu and X. Wan, "Ppsgen: learning to generate presentation slides for academic papers," in *Proceedings of the Twenty-Third international joint conference on Artificial Intelligence*. AAAI Press, 2013, pp. 2099–2105.

[18] Z. Cao, F. Wei, L. Dong, S. Li, and M. Zhou, "Ranking with recursive neural networks and its application to multi-document summarization," in *Proceedings of the 2015 AAAI Conference on Artificial Intelligence*. Association for the Advancement of Artificial Intelligence, 2015.

[19] Y. Bengio, R. Ducharme, P. Vincent, and C. Janvin, "A neural probabilistic language model," *The Journal of Machine Learning Research*, vol. 3, pp. 1137–1155, 2003.

[20] Y. Bengio, H. Schwenk, J.-S. Senécal, F. Morin, and J.-L. Gauvain, "Neural probabilistic language models," in *Innovations in Machine Learning*. Springer, 2006, pp. 137–186.

[21] R. Collobert and J. Weston, "A unified architecture for natural language processing: Deep neural networks with multitask learning," in *Proceedings of the 25th international conference on Machine learning*. ACM, 2008, pp. 160–167.

[22] A. Mnih and G. E. Hinton, "A scalable hierarchical distributed language model," in *Advances in neural information processing systems*, 2009, pp. 1081–1088.

[23] J. Turian, L. Ratinov, and Y. Bengio, "Word representations: a simple and general method for semi-supervised learning," in *Proceedings of the 48th annual meeting of the association for computational linguistics*. Association for Computational Linguistics, 2010, pp. 384–394.

[24] T. Mikolov, K. Chen, G. Corrado, and J. Dean, "Efficient estimation of word representations in vector space," *arXiv preprint arXiv:1301.3781*, 2013.

[25] T. Mikolov, I. Sutskever, K. Chen, G. S. Corrado, and J. Dean, "Distributed representations of words and phrases and their compositionality," in *Advances in neural information processing systems*, 2013, pp. 3111–3119.

[26] F. M. Zanzotto, I. Korkontzelos, F. Fallucchi, and S. Manandhar, "Estimating linear models for compositional distributional semantics," in *Proceedings of the 23rd International Conference on Computational Linguistics*. Association for Computational Linguistics, 2010, pp. 1263–1271.

[27] A. Yessenalina and C. Cardie, "Compositional matrix-space models for sentiment analysis," in *Proceedings of the Conference on Empirical Methods in Natural Language Processing*. Association for Computational Linguistics, 2011, pp. 172–182.

[28] R. Socher, C. C. Lin, C. Manning, and A. Y. Ng, "Parsing natural scenes and natural language with recursive neural networks," in *Proceedings of the 28th international conference on machine learning (ICML-11)*, 2011, pp. 129–136.

[29] E. Grefenstette, G. Dinu, Y.-Z. Zhang, M. Sadrzadeh, and M. Baroni, "Multi-step regression learning for compositional distributional semantics," *arXiv preprint arXiv:1301.6939*, 2013.

[30] G.-B. Huang, Q.-Y. Zhu, and C.-K. Siew, "Extreme learning machine: theory and applications," *Neurocomputing*, vol. 70, no. 1, pp. 489–501, 2006.

[31] R. J. Williams, D. E. Rumelhart, and G. E. Hinton, "Learning representations by back-propagating errors," *Nature*, pp. 323–533, 1986.

[32] D. E. Rumelhart, G. E. Hinton, and R. J. Williams, "Learning representations by back-propagating errors," *Cognitive modeling*, vol. 5, p. 3, 1988.

[33] J. B. Pollack, "Recursive distributed representations," *Artificial Intelligence*, vol. 46, no. 1, pp. 77–105, 1990.

[34] J. L. Elman, "Distributed representations, simple recurrent networks, and grammatical structure," *Machine learning*, vol. 7, no. 2-3, pp. 195–225, 1991.

[35] E. H. Huang, R. Socher, C. D. Manning, and A. Y. Ng, "Improving word representations via global context and multiple word prototypes," in *Proceedings of the 50th Annual Meeting of the Association for Computational Linguistics: Long Papers-Volume 1*. Association for Computational Linguistics, 2012, pp. 873–882.

[36] W. Y. Zou, R. Socher, D. M. Cer, and C. D. Manning, "Bilingual

word embeddings for phrase-based machine translation." in *EMNLP*, 2013, pp. 1393–1398.

[37] R. Socher, E. H. Huang, J. Pennin, C. D. Manning, and A. Y. Ng, "Dynamic pooling and unfolding recursive autoencoders for paraphrase detection," in *Advances in Neural Information Processing Systems*, 2011, pp. 801–809.

[38] R. Socher, J. Pennington, E. H. Huang, A. Y. Ng, and C. D. Manning, "Semi-supervised recursive autoencoders for predicting sentiment distributions," in *Proceedings of the Conference on Empirical Methods in Natural Language Processing*. Association for Computational Linguistics, 2011, pp. 151–161.

[39] R. Socher, D. Chen, C. D. Manning, and A. Ng, "Reasoning with neural tensor networks for knowledge base completion," in *Advances in Neural Information Processing Systems*, 2013, pp. 926–934.

[40] R. Socher, A. Perelygin, J. Y. Wu, J. Chuang, C. D. Manning, A. Y. Ng, and C. Potts, "Recursive deep models for semantic compositionality over a sentiment treebank," in *Proceedings of the conference on empirical methods in natural language processing (EMNLP)*, vol. 1631. Citeseer, 2013, p. 1642.

[41] Q. V. Le and T. Mikolov, "Distributed representations of sentences and documents," *arXiv preprint arXiv:1405.4053*, 2014.

[42] G.-B. Huang, "What are extreme learning machines? filling the gap between frank rosenblatts dream and john von neumanns puzzle," *Cognitive Computation*, vol. 7, no. 3, pp. 263–278, 2015.

[43] C. R. Rao and S. K. Mitra, *Generalized inverse of matrices and its applications*. Wiley New York, 1971, vol. 7.

[44] N.-Y. Liang, G.-B. Huang, P. Saratchandran, and N. Sundararajan, "A fast and accurate online sequential learning algorithm for feedforward networks," *Neural Networks, IEEE Transactions on*, vol. 17, no. 6, pp. 1411–1423, 2006.

[45] G. Huang, G.-B. Huang, S. Song, and K. You, "Trends in extreme learning machines: A review," *Neural Networks*, vol. 61, pp. 32–48, 2015.

[46] K. Owczarzak, J. M. Conroy, H. T. Dang, and A. Nenkova, "An assessment of the accuracy of automatic evaluation in summarization," in *Proceedings of Workshop on Evaluation Metrics and System Comparison for Automatic Summarization*. Association for Computational Linguistics, 2012, pp. 1–9.

[47] G.-B. Huang, H. Zhou, X. Ding, and R. Zhang, "Extreme learning machine for regression and multiclass classification," *Systems, Man, and Cybernetics, Part B: Cybernetics, IEEE Transactions on*, vol. 42, no. 2, pp. 513–529, 2012.

[48] P. L. Bartlett, "The sample complexity of pattern classification with neural networks: the size of the weights is more important than the size of the network," *Information Theory, IEEE Transactions on*, vol. 44, no. 2, pp. 525–536, 1998.

[49] Y. Li and S. Li, "Query-focused multi-document summarization: Combining a topic model with graph-based semi-supervised learning," in *Proceedings of COLING 2014, the 25th International Conference on Computational Linguistics: Technical Papers*, 2014, p. 11971207.

[50] C.-Y. Lin, "Rouge: A package for automatic evaluation of summaries," in *Text Summarization Branches Out: Proceedings of the ACL-04 Workshop*, 2004, pp. 74–81.

[51] Z. He, C. Chen, J. Bu, C. Wang, L. Zhang, D. Cai, and X. He, "Document summarization based on data reconstruction." in *AAAI*, 2012.