 CMC, vol.55, no.2, pp.213-227, 2018

Paragraph Vector Representation Based on Word to Vector and CNN Learning

Zeyu Xiong[1, *], Qiangqiang Shen[1], Yijie Wang[1] and Chenyang Zhu[2]

Abstract: Document processing in natural language includes retrieval, sentiment analysis, theme extraction, etc. Classical methods for handling these tasks are based on models of probability, semantics and networks for machine learning. The probability model is *loss of semantic information* in essential, and it influences the processing accuracy. Machine learning approaches include supervised, unsupervised, and semi-supervised approaches, labeled corpora is necessary for semantics model and supervised learning. The method for achieving a reliably labeled corpus is done manually, it is *costly and time-consuming* because people have to read each document and annotate the label of each document. Recently, the continuous CBOW model is efficient for learning high-quality distributed vector representations, and it can capture a large number of precise syntactic and semantic word relationships, this model can be easily extended to learn paragraph vector, but it *is not precise*. Towards these problems, this paper is devoted to developing a new model for learning paragraph vector, we combine the CBOW model and CNNs to establish a new deep learning model. Experimental results show that paragraph vector generated by the new model is better than the paragraph vector generated by CBOW model in semantic relativeness and accuracy.

Keywords: Distributed word vector, distributed paragraph vector, CNNs, CBOW, deep learning.

1 Introduction

There are many unstructured data existed as an important source for data analysis. Text classification, text mining, text clustering, sentiment analysis, automatic speech recognition and machine translation are become as important tasks in information retrieval and natural language processing.

In actual application, raw documents need to be transformed into numerical vectors, in which each document's hidden characteristics are captured. Distributed representations of words in a vector space help machine learning algorithms to achieve better performance in processing various IR and NLP tasks. One of the earliest application to word vector dates back to 1986 due to Rumelhart et al. [Rumelhart, Hinton and Williams (1986)], has become a successful paradigm, especially for statistical language modeling [Elman (1990); Bengio,

[1] College of Computer, National University of Defense Technology, De Ya Road, Changsha 410073, China.

[2] School of Computer Science, Simon Fraser University, 8888 UNIVERSITY DRIVE, BURNABY, BC, V5A 1S6, Vancouver, Canada.

[*]Corresponding author: Zeyu Xiong. Email: xiongzeyu08@nudt.edu.cn.

Schwenk, Senécal et al. (2006); Mikolov (2012)], text mining tasks [Huang (2008); Wu, SC and Yu (2010)].

Various neural networks are used in learning vector representations of words [Bengio, Schwenk, Senécal et al. (2006); Collobert and Weston (2008); Mnih and Hinton (2008); Mikolov, Sutskever, Chen et al. (2013)]. In the formulation of word vector, each word is represented by a vector which is initially concatenated or averaged with other word vectors in a context, and the final resulting vector is obtained to predict other words in the context. For example, Bengio et al. [Bengio, Schwenk, Senécal et al. (2006)] use the proposed neural network language model to form the input of a neural network, and use the resulting vector to predict the next word. The training outcome to the neural network model is that the word vectors are mapped into a vector space such that semantically similar words have similar vector representations (e.g. vec ("Madrid")–vec ("Spain")+vec ("France") is closer to vec ("Paris")). Recently, Mikolov et al. [Mikolov, Sutskever, Chen et al. (2013)] introduced the CBOW model which have high quality vector representations of words by learning from large amounts of unstructured text data, and the training processing of the CBOW model does not involve dense matrix multiplications.

Word of vector has been used to represent document in a matrix form, also can be used to generate the vector space model. In the vector space model, each document is represented as a vector with one real-valued component, usually a *tf-idf* weight for each term. Kim et al. [Kim, Kim and Cho (2017)] create concepts through clustering word vectors generated from word2vec, and use the frequencies of these concept clusters to represent document vectors. Huang [Huang 2008] uses word of vector for solving text clustering task. Tsai et al. [Tsai, Wang and Chien (2016)] establish an approach to discovering financial keywords from a large number of financial reports by applying the continuous bag-of-words (CBOW) model. Cao et al. [Cao, Zhou, Sun et al. (2018)] use the histogram of the bag of words model (BOW) to determine the number of sub-images in the image that convey secret information for the purpose of improving the retrieval efficiency, Feldman [Feldman (2013)] analyses techniques and applications of word of vector for sentiment analysis.

Word representations are limited in ability for representing idiomatic phrases that are not compositions of the individual words. For example, "Reference News" is a newspaper, and it is not a natural combination of the meanings of "Reference" and "News". The continuous Skip-gram model and CBOW model both are efficient method for learning high-quality distributed vector representations that capture a large number of precise syntactic and semantic word relationships. Therefore, using vectors to represent the whole phrases makes the Skip-gram or CBOW model considerably more expressive. Mikolov et al. [Mikolov, Sutskever, Chen et al. (2013)] develop a simple method for finding phrases in text, and show that learning good vector representations for millions of phrases is possible. Other methods that aim to represent meaning of sentences by composing the word vectors, such as the recursive auto encoders [Socher, Pennington, Huang et al. (2011)], is also originated by using phrase vectors instead of the word vectors.

Paragraph vector is constructed in Mikolov et al. [Mikolov, Sutskever, Chen et al. (2013)] and Le et al. [Le and Mikolov (2014)] by an unsupervised algorithm that learns fixed-length feature representations from variable-length pieces of texts, such as sentences, paragraphs, and documents. Both word vectors and paragraph vectors are trained by the

stochastic gradient descent and back-propagation. Paragraph vectors are unique among paragraphs, and the word vectors are asked to be shared. But, we find that this sharing of word vectors in different paragraph is not well kept during the training processing. It inspirits us to establish a new neural network to generate paragraph vectors and keep the word vectors shared. Yuan et al. [Yuan, Li, Wu et al. (2017)] use convolutional neural network and principal component analysis to detect fingerprint liveness from different fingerprint materials. Therefore, a hybrid neural networks by the combination of CBOW and CNN is proposed in this paper.

The rest of this paper is organized as follows: Section 2 introduces prior research related to one-hot representation for word, Hierarchical softmax model, convolutional neural networks. Section 3 formally presents our approach in combination of CBOW and CNN model for paragraph vector learning. Section 4 describes the experimental settings and experimental results. At last, we conclude the paper and discuss some future work in Section 5.

2 Some related works

2.1 One-hot representation for word and compositional vector model for sentence

Many machine learning algorithms cannot handle on label data directly. They require all input variables and output variables to be numerical form. In the following, we briefly describe the one-hot representation for word and compositional vector model for sentence.

Let $\mathrm{V} = \{w_1, w_2, \cdots, w_{|V|}\}$ be a lexicon, each word of $w \in V$ is represented by a one-hot encoded vector $v(w)$ of $|V|$ dimensions, which means to be 1 at the indexing position of w, and all other $|V| - 1$ indexing positions are 0. One-hot document vector is generated similarly (see Fig. 1).

[Document] Are you looking for antivirus software for your Apple computer but don't know who to trust? Or are you unsure if your antivirus is the right choice? These days, having trusted antivirus software is an important part of life. With recent media concerns over personal data security, and numerous harmful viruses effecting thousands of computer owners worldwide, it's important you choose the right provider. There is no need to worry, our Mac specialist team have reviewed the most popular providers and ranked the most trusted 1-10. Our goal is to make sure your computer, and personal data are kept safe!

	v(1):Are	v(2):you	v(3):looking	v(4):for	v(5):antivirus	v(6):software	…
Document	3	3	1	2	3	2	…

Figure 1: One-hot document vector via bag of words

Compositional vector representation for words has proven very efficient for many NLP tasks. One common method for generating a vector of a sentence is to average all the vectors of words in this sentence. Mitchell et al. [Mitchell and Lapata (2008)] use the

combination rules with operators: Addition and multiplication to generate a higher-level representation for a sentence/document. Some more complex composition functions using parsed tree, matrix-vector composition, convolutional neural networks or tensor composition can be seen in the above issues [Socher, Lin, Ng et al. (2011); Socher, Huval, Manning et al. (2013); Hermann and Blunsom (2013); Tsubaki, Duh, Shimbo et al. (2013)]. Some of these works use deep linguistic structures as parsed trees to design their composition functions, and Pham and Le [Pham and Le (2018)] employ other semantic signals such as sentiment or topic labels for designing the objective functions. Xiong et al. [Xiong and Wang (2018)] develop a new method for calculating the field weights based on the ranking score.

2.2 Hierarchical softmax for CBOW model

CBOW model is one of word embedding models, it represents words with real-valued vectors whose relative similarities correlate with semantic similarity. It can be used for computing similarities between terms.

The training objective for CBOW model (Fig. 2) is to learn nearby words vector representation that are good at predicting the center word.

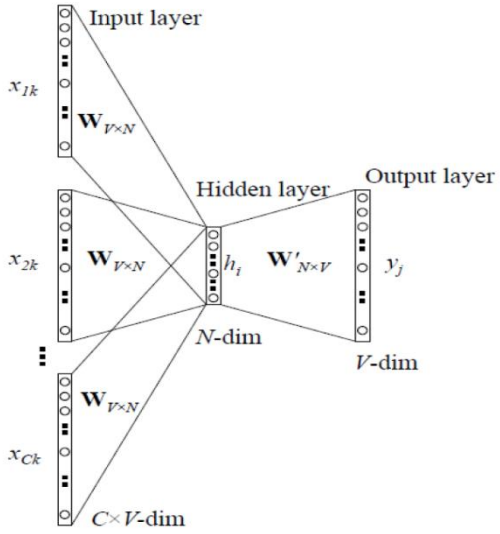

Figure 2: The architecture for CBOW model [Mikolov, Sutskever, Chen et al. (2013)]

Given a sequence of training words: $w_1, w_2, \cdots, w_T$, the objective of the CBOW model is to maximize the average log probability

$$\frac{1}{T}\sum_{t=1}^{T}\sum_{-c\le j\le c, j\neq 0}\log p(w_{t+j}|w_t) \quad (1)$$

where c is the size of the training context, larger c results in more training examples and thus can lead to a higher accuracy, and cost more training time. The basic CBOW formulation defines $\mathrm{p}(w_{t+j}|w_t)$ by using the following full softmax function:

$$p(w_{t+j}|w_t) = \frac{\exp((V'_{t+j})^T V_{w_{t+j}})}{\sum_{i=1}^{|V|} \exp((V'_i)^T V_{w_{i+j}})} \quad (2)$$

where V_{w_t} is input vector representation of word w_t, and $(V'_{t+j})^T, (V'_i)^T$ are output vector representations of words w_{t+j}, w_i.

A computationally efficient approximation of the full softmax is the hierarchical softmax. The hierarchical softmax uses a binary tree representation of the output layer with the $|V|$ words as its leaves and, for each word *w* located at the leaf, let $n(w,j)$ be the *j*-th node on the path from the root to *w*, and let $Len(w)$ be the length of this path, so $n(w,1) = root$ and $n(w, len(w)) = w$. Let $child(n)$ be an arbitrary fixed child of inner node *n*, and let $\langle x \rangle$ be 1 if *x* is true and -1, otherwise, then the hierarchical softmax is defined as follows:

$$p(\mathrm{w}|w_I) = \prod_{j=1}^{Len(w)} \sigma(\langle n(w,j+1) = child(n(w,j)) \rangle . (V'_{n(w,j)})^T V_{w_I}) \quad (3)$$

where $\sigma(x) = 1/(1+\exp(-x))$, the cost of computing $\log(p(w_0|w_I))$ and $\nabla \log p(w_0|w_I)$ is proportional to $Len(w_0)$, which on average is no greater than $\log w$.

2.3 Paragraph vector representation

Le et al. [Le and Mikolov (2014)] has established paragraph vector framework, which is an unsupervised manner that learns continuous distributed vector representations for pieces of texts. The pieces of texts can be of variable-length, includes phrase, sentence, paragraph and document. Using the name *Paragraph Vector* is to insist the fact that the method can be applied to variable-length pieces of texts, range from a phrase or sentence to a large document.

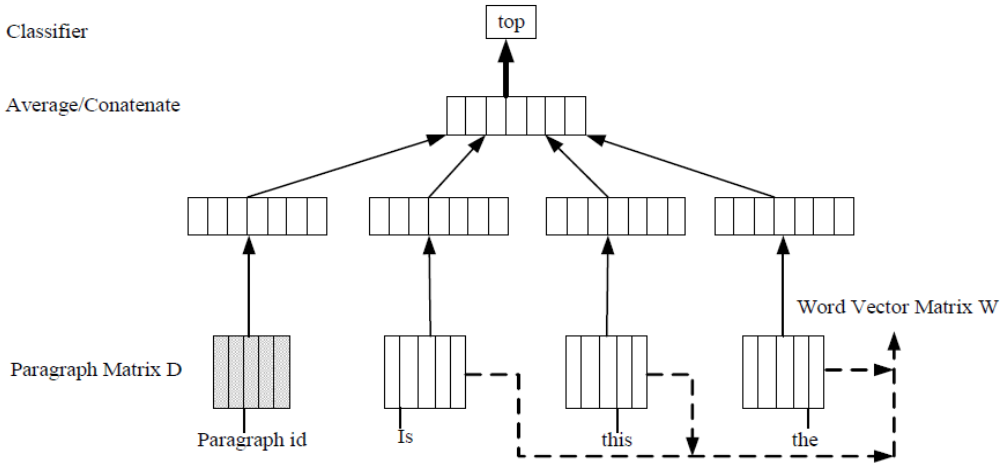

Figure 3: A framework for learning paragraph vector [Le and Mikolov (2014)]

The paragraph vector framework actually is a distributed memory model (Fig. 3), every paragraph is mapped to a unique vector, represented by a column in matrix *D* and every word is also mapped to a unique vector, represented by a column in matrix *W*. In this model, the concatenation or average of this vector with a context of three words ("is", "this", "the")

is used to predict the fourth word ("top").

2.4 Convolutional neural networks

Convolutional neural networks also called as CNNs or CNN [LeCun (1989)], is a specialized kind of neural network for processing data that has a known, grid-like topology. Convolutional networks are simply neural networks that use convolution to replace general matrix multiplication with at least one of their network layers.

In order to employ CNNs availably, we need to understand the motivation behind using convolution in a neural network. Convolution leverages three important ideas that can help improve a machine learning system: *Sparse interactions*, *parameter sharing* and *equivariant representations* [Goodfellow, Bengio and Courville (2016)]. Sparse interactions is accomplished by making the kernel smaller than the input, parameter sharing refers to using the same parameter for more than one function in a model, in the case of convolution, the particular form of parameter sharing causes the layer to have a property called equivariance to translation. One of the basic operation in CNNs is called *pooling*, which almost all convolutional networks use.

3 The CNN model for paragraph vector representation

3.1 Motivation: Example of the topic analysis to actual paragraph.

Consider the paragraph stated as follows [ABC news, Feb 7, 2018]:

Figure 4: Relative words marked to theme in a paragraph

Fig. 4 marked the relative words to express the theme information in a paragraph, the relative terms to express theme are marked same color in word2vec space, which words with same color are at nearby position in a paragraph. We have known that CNNs have achieved great success for acquiring local neighboring feature (for example: Finger, edge etc.) in image processing, if we let a word in a paragraph viewed as a pixel in a document, we can use CNNs to generate feature of theme in a paragraph, and this feature is closed relative to paragraph vector.

3.2. Network architecture

We design the CNNs for learning paragraph vector (Fig. 5), which consists of three convolutional layers and one fully connected layer, the input word vectors in a paragraph come down from word2vec (CBOW), is a 56×50 matrix, and the output layer is a 1×50 vector as a paragraph vector.

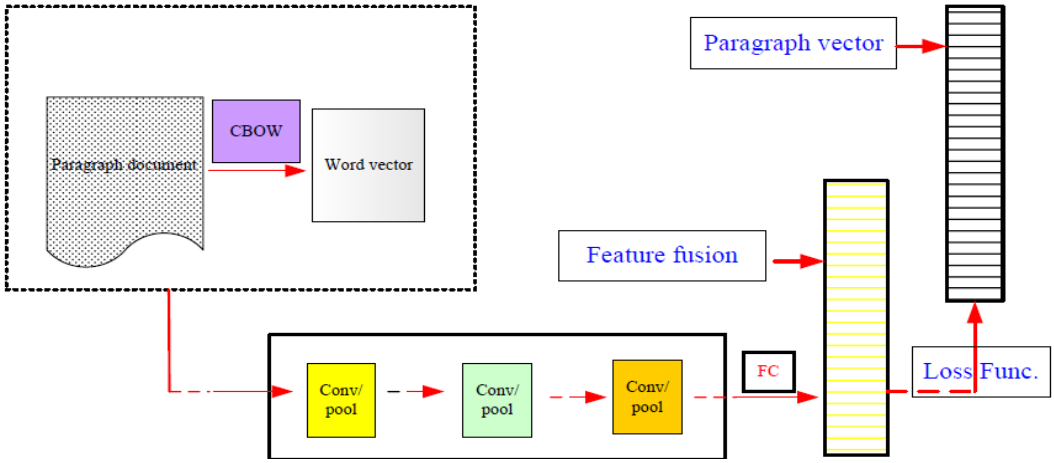

Figure 5: The frame for generating paragraph vector

The parameters in Fig. 5 are listed in Tab. 1. Behind the three convolutional layers is a flatten layer, which is a fully connected layer and perform feature fusion, and finally 50 neurons are set up.

Table 1: Parameters in CNNs

	filters	**size of kernel**	**strides**	**Activation**
Layer1	10	5	2	Relu
Layer2	20	5	2	Relu
Layer3	40	5	2	Relu

4 Experiments

4.1 Dataset described

We use the Stanford Sentiment *Treebank* dataset. Every example in this dataset is a single sentence, and every sentence in the dataset has a label which goes from very negative to very positive in the scale from 0.0 to 1.0. The labels are generated by human annotators using Amazon Mechanical Turk. We extracted 10,000 paragraphs from the dataset, so each training process needs to input 56×50 matrix which corresponds to each paragraph.

4.2 Result analysis

In order to verify the availability of the generated paragraph vector for new model, we compare with the model CBOW designed by Le et al. [Le and Mikolov (2014)]. According to the distribution property in vector space, paragraph vector should be in a closer neighborhood of mean vector in a paragraph, the distance behavior performs indirectly semantic relativeness and accuracy in vector space. Therefore, we introduce the measure of distance between paragraph vector and mean of word vectors in a paragraph, Fig. 6 to Fig. 15 shows the contrast results for each 1000 paragraphs, the blue curve is plotted for CBOW model and the red curve is plotted for CNNs.

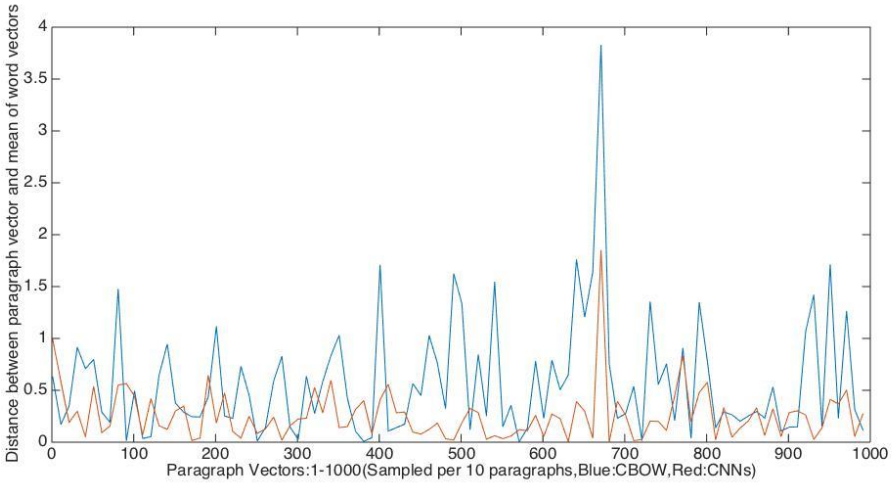

Figure 6: Distance between paragraph vector and mean of word vectors in paragraphs 1-1000

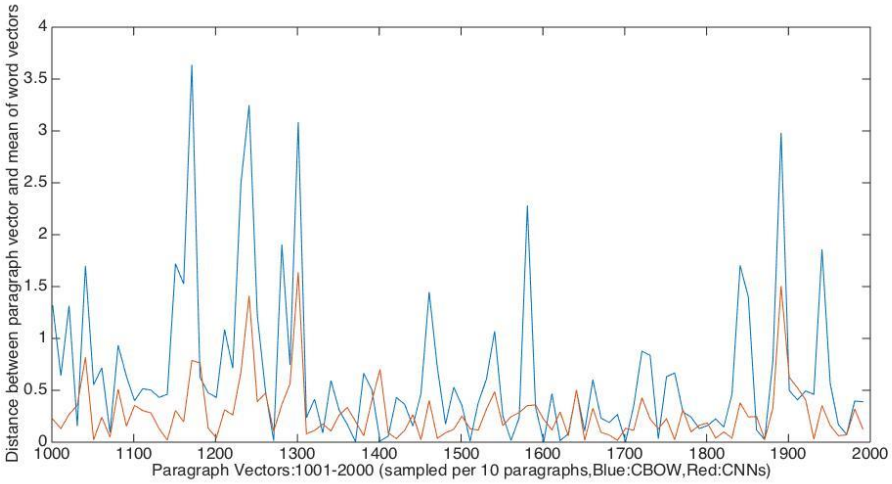

Figure 7: Distance between paragraph vector and mean of word vectors in paragraphs 1001-2000

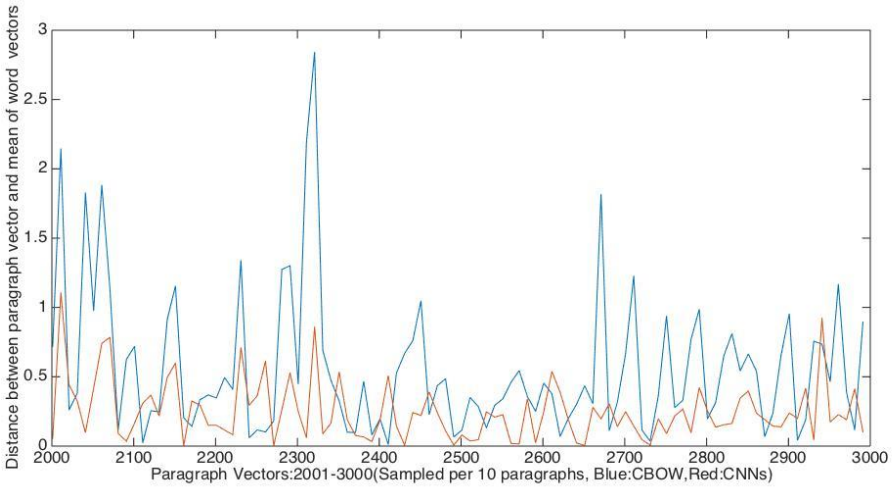

Figure 8: Distance between paragraph vector and mean of word vectors in paragraphs 2001-3000

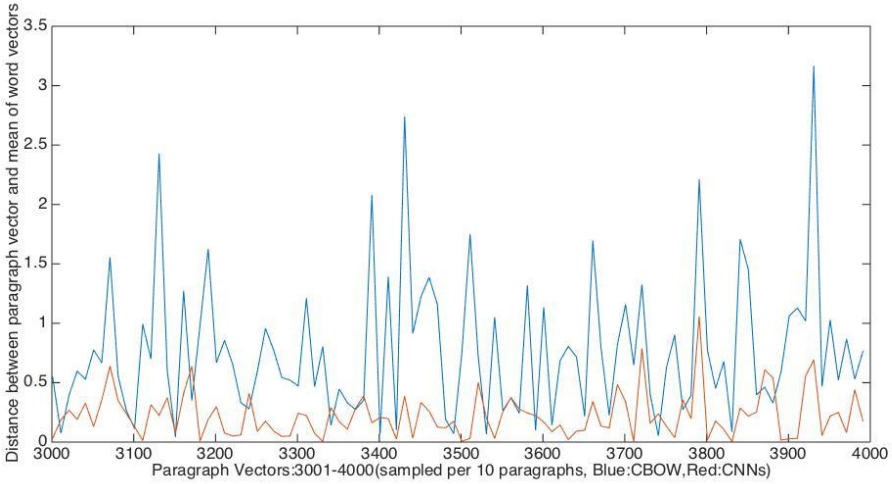

Figure 9: Distance between paragraph vector and mean of word vectors in paragraphs 3001-4000

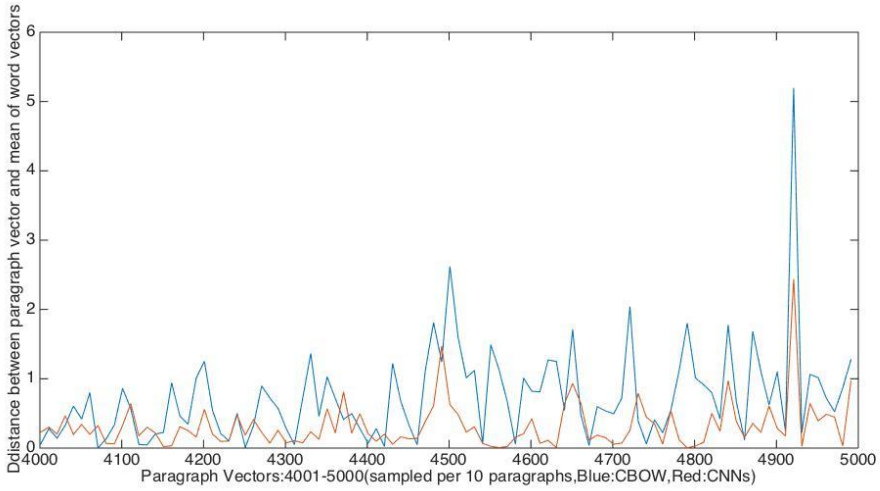

Figure 10: Distance between paragraph vector and mean of word vectors in paragraphs 4001-5000

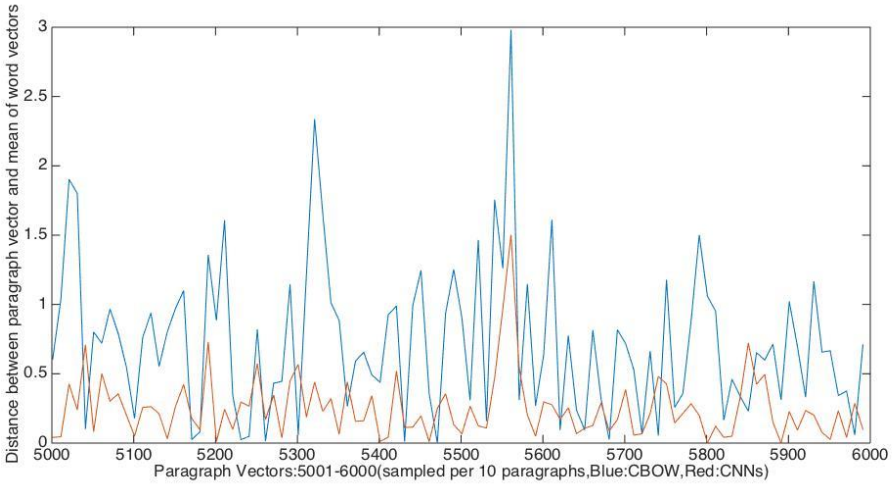

Figure 11: Distance between paragraph vector and mean of word vectors in paragraphs 5001-6000

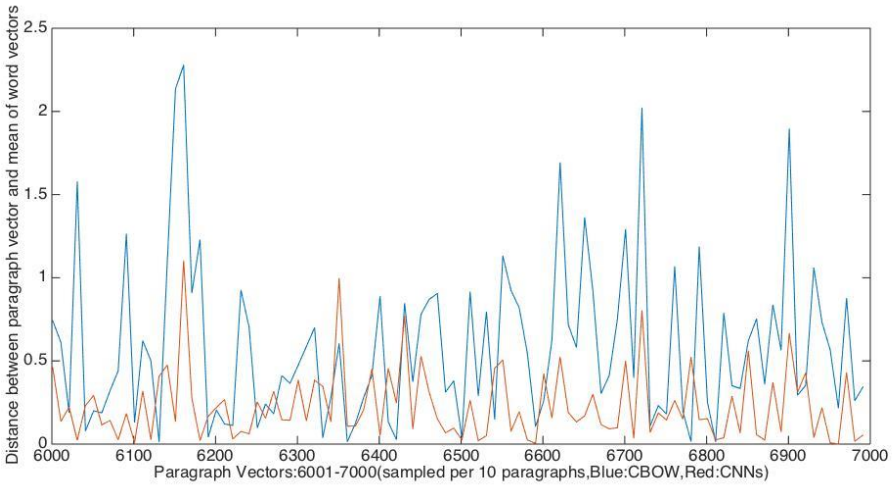

Figure 12: Distance between paragraph vector and mean of word vectors in paragraphs 6001-7000

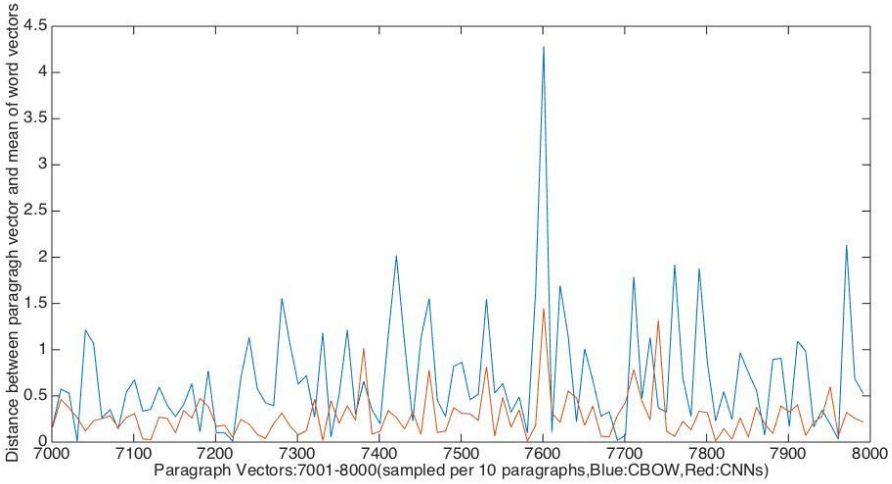

Figure 13: Distance between paragraph vector and mean of word vectors in paragraphs 7001-8000

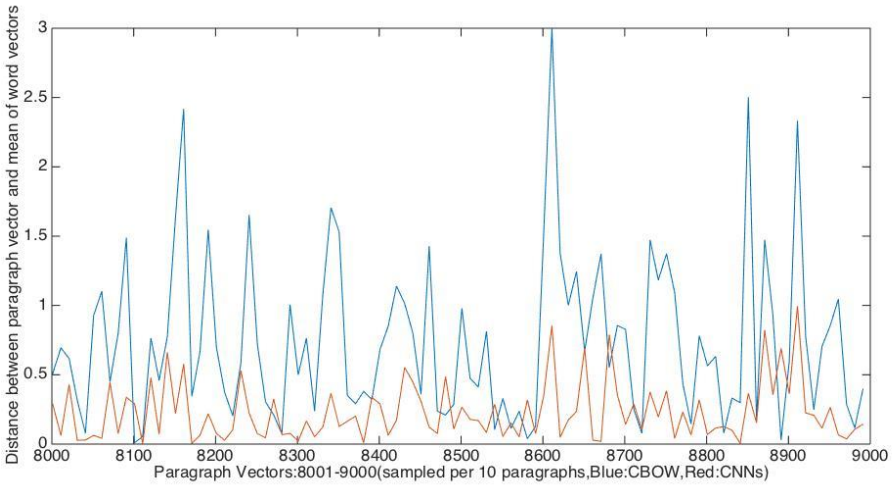

Figure 14: Distance between paragraph vector and mean of word vectors in paragraphs 8001-9000

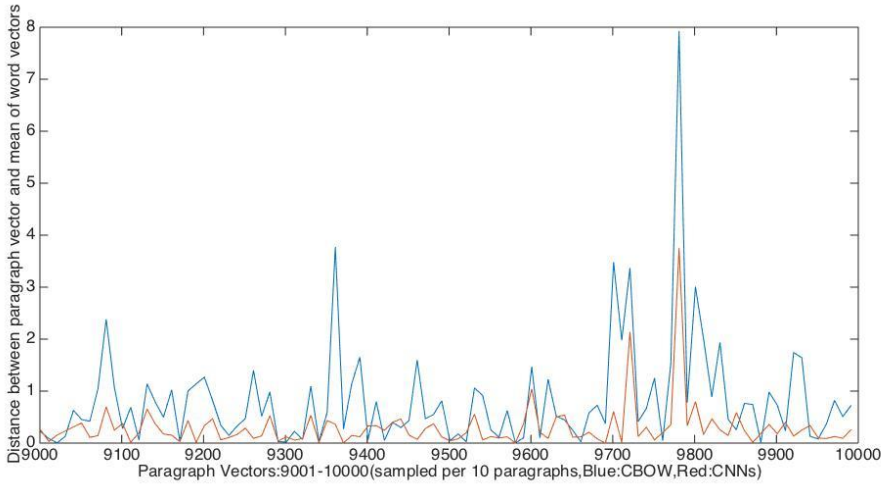

Figure 15: Distance between paragraph vector and mean of word vectors in paragraphs 9001-10000

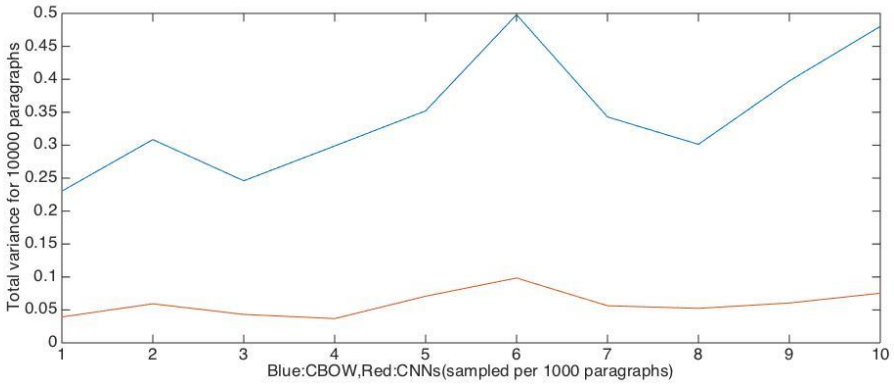

Figure 16: The total variance sampled from 1-10000 paragraphs

Fig. 16 shows the result of total variance for sampled 10000 paragraphs.

It is easy to see from Fig. 6 to Fig. 16 that the constructed deep learning network (CNNs in Fig. 5) achieves better availability in distance measure and total variance behavior. The comparative results show that the generated paragraph vector in this paper is in a closer neighborhood of mean vector in a paragraph, the measure distance performs indirectly semantic relativeness and accuracy in vector space.

5 Conclusions

The contrast experimental results show that paragraph vectors generated by combined CBOW model and CNNs model achieve more available result than the CBOW model did, this new model is also an unsupervised algorithm that learns fixed-length feature representations from variable-length pieces of texts.

The well distributed representation of paragraph vector is benefitted for machine learning and deep learning, but well qualified paragraph vector is difficult to generate. As we seen, a paragraph generally contain several sentences, and a sentence usually contain several words, therefore, well sentence vector should be constructed firstly, we may try to use our new model in this paper to generate sentence vector and paragraph vector simultaneously.

Both CBOW and Skip-gram are distributed word vector representation, in order to learn semantics to one center word, we choose CBOW and CNNs to generate paragraph vector, we may try to use combined Skip-gram and CNNs to produce paragraph vector, further analysis will be given to explain advantages for these two hybrid models respectively.

Another open problem is that what is the good criterion to verify paragraph vector well or not higher level semantics learning is necessary, we expected to construct deeper network to learn paragraph vector.

Future work for us is to use our paragraph vector dealing with some complex NLP (Natural Language Processing) tasks in directive manner, for example, task of sentiment classification, task of theme extracting, etc.

Acknowledgement: The authors would like to thank all anonymous reviewers for their suggestions and feedback. This work Supported by the National Natural Science, Foundation

of China (No. 61379052, 61379103), the National Key Research and Development Program (2016YFB1000101). The Natural Science Foundation for Distinguished Young Scholars of Hunan Province (Grant No. 14JJ1026), Specialized Research Fund for the Doctoral Program of Higher Education (Grant No. 20124307110015).

References

Bengio, Y.; Schwenk, H.; Sen é cal, J.; Morin, F.; Gauvain, J. (2006): Neural probabilistic language models. Innovations in Machine Learning. *Springer*, pp. 137-186.

Cao, Y.; Zhou, Z; Sun, X.; Gao, C. (2018): Coverless information hiding based on the molecular structure images of material. *Computers, Materials & Continua*, vol. 54, no. 2, pp. 197-207.

Collobert, R.; Weston, J. (2008): A unified architecture for natural language processing: Deep neural networks with multitask learning. *Proceedings of the 25th International Conference on Machine Learning*, pp. 160-167.

Elman, J. (1990): Finding structure in time. *Cognitive Science*, pp. 179-211.

Feldman, R. (2013): Techniques and applications for sentiment analysis. *Communications of the ACM*, vol. 56, no. 4, pp. 82-89.

Goodfellow, I.; Bengio, Y.; Courville, A. (2016): *Deep Learning*. www.deeplearningbook.org.

Hermann, K. M.; Blunsom, P. (2013): The role of syntax in vector space models of compositional semantics. *Proceedings of ACL*, pp. 894-904.

Huang, A. (2008): Similarity measures for text document clustering. *Proceedings of the Sixth New Zealand Computer Science Research Student Conference*, pp. 49-56.

Kim, H. K.; Kim, H.; Cho, S. (2017): Bag-of-concepts: Comprehending document representation through clustering words in distributed representation. *Neurocomputing*, vol. 266, no. 2017, pp. 336-352.

Le, Q.; Mikolov, T. (2014): *Distributed representations of sentences and documents*.

LeCun, Y. (1989): Generalization and network design strategies. *Technical Report CRG-TR-89-4*, University of Toronto.

Mikolov, T. (2012): *Statistical language models based on neural networks. (Ph.D. Thesis)*. Brno University of Technology, the Czech Republic.

Mikolov, T.; Sutskever, I.; Chen, K.; Corrado, G.; Dean, J. (2013): *Distributed representations of words and phrases and their compositionality*.

Mitchell, J.; Lapata, M. (2008): Vector-based models of semantic composition. *Proceedings of ACL*, pp. 236-244.

Mnih, A.; Hinton, G. (2008): A scalable hierarchical distributed language model. *Advances in Neural Information Processing Systems*, pp. 1081-1088.

Pham, D. H.; Le, A. C. (2018): Learning multiple layers of knowledge representation for aspect based sentiment analysis. *Data & Knowledge Engineering*, vol. 114, no. 2018, pp. 26-39.

Rumelhart, D. E.; Hintont, G. E.; Williams R. J. (1986): Learning representations by

back propagating. *Nature*, vol. 323, no. 6088, pp. 533-536.

Socher, R.; Huval, B.; Manning, C. D.; Ng, A. Y. (2013): Semantic compositionality through recursive matrix-vector spaces. *Proceedings of EMNLP-CoNLL*, pp. 1201-1211.

Socher, R.; Lin, C.; Ng, A.; Manning C. (2011): Parsing natural scenes and natural language with recursive neural networks. *Proceedings of the 26th International Conference on Machine Learning*, pp. 129-136.

Socher, R.; Pennington, J.; Huang, E. H.; Ng, A. Y.; Manning, C. D. (2011): Semi-supervised recursive autoencoders for predicting sentiment distributions. *Proceedings of EMNLP*, pp. 151-161.

Tsai, M. F.; Wang, C.; Chien, P. C. (2016): Discovering finance keywords via continuous-space language models. *ACM Transactions on Management Information Systems*, vol. 7, no. 3.

Tsubaki, M.; Duh, K.; Shimbo, M.; Matsumoto, Y. (2013): Modeling and learning semantic co-compositionality through prototype projections and neural networks. *Conference on Empirical Methods in Natural Language Processing*, pp. 130-140.

Wu, L.; SC, H.; Yu, N. (2010): Semantics-preserving bag-of-words models and applications. *IEEE Transactions on Image Processing*, vol. 19, no. 7, pp. 1908-1920.

Xiong, Z.; Wang, Y. (2018): New document scoring model based on interval tree. *Journal of Visual Languages and Computing*, vol. 45, no. 2018, pp. 39-143.

Yuan, C.; Li, X.; Wu, Q.; Li, J.; Sun, X. (2017): Fingerprint liveness detection from different fingerprint materials using convolutional neural network and principal component analysis. *Computers, Materials & Continua*, vol. 53, no. 4, pp. 357-372.