Learning Task Specific Distributed Paragraph Representations Using a Two-Tier Convolutional Neural Network

Tao Chen[1], Ruifeng Xu[1], Yulan He[2], Xuan Wang[1]
[1]Shenzhen Engineering Laboratory of Performance Robots at Digital Stage
Shenzhen Graduate School, Harbin Institute of Technology, China
[2]School of Engineering and Applied Science, Aston University, UK

Abstract. We introduce a type of two-tier convolutional neural network model for learning distributed paragraph representations for a special task (e.g. topic classification or sentiment classification). We decompose the paragraph semantics into three cascaded constitutes: word representation, sentence composition and document composition. Specifically, we learn distributed word representations by a continuous bag-of-words model from a large unstructured text corpus. Then, using these word representations as pre-trained vectors, the first tier of our model learns the distributed task-specific sentence representations from a corpus where each sentence is annotated with a task-specific label. Subsequently, the second tier of our model learns the distributed paragraph representations of a different document corpus from the learned sentence representations. Our proposed model has been evaluated on topic classification based on the DBpedia ontology and sentiment classification of Amazon reviews. Empirical results show the effectiveness of our proposed learning model for generating distributed paragraph representations.

Keywords: Natural language processing, distributed representation, convolutional neural network.

1 Introduction

Paragraph or short document representations are important for a variety of NLP tasks including text classification, sentiment analysis and document retrieval. Many applications use distributional representation of text, such as Bag-Of-Words (BOW) representation, as the input to their algorithms. With the rapid development of deep neural networking and parallel computing, state-of-the-art results have been achieved in many NLP tasks using the distributed representations of word generated by deep learning [1, 3, 12, 17, 7, 8]. While distributed representations can be learned at the word level, they can also be learned at higher semantic levels such as phrases, paragraphs or documents.

The distributed representation of a paragraph usually refers to dense and real-valued vectors in a low-dimensional space to represent the paragraph, which are assumed to convey semantic information contained in it. Many existing approaches learn distributed paragraph representations without incorporating the knowledge of text structure or task-specific labels (e.g. sentiment labels for the sentiment classification task). For example, Le and Mikolov [9] proposed an unsupervised framework to learn continuous distributed vector representations for pieces of texts ranging from sentences to documents based on the context of words in a large unlabeled corpus. Zhang and LeCun [18] employed a nine-layer deep temporal convolutional networks to learn representations from character-level inputs all the way up to abstract text concepts without any syntactic or semantic structures. Dos Santos and Gatti [16] proposed a deep convolutional neural network (CNN) that also exploits sentence-level information directly from character-level information.

We argue that for supervised learning tasks, the knowledge of text structures and the task-specific annotation information could be potentially useful for improving the end performance. Note that such supervised information could be obtained elsewhere rather than the target data corpus for supervised learning tasks. For example, for sentiment classification on Amazon reviews, the review dataset only contains sentiment labels at the document level. But we can obtain the sentence-level sentiment labels from other datasets such as the Sentiment Treebank. We propose a supervised framework that learns distributed paragraph representations built hierarchically from words, sentences and paragraphs with the incorporation of supervised information. Our framework works as follows. Firstly, distributed word representations are learned by a Continuous Bag-Of-Words (CBOW) model [12] from a large unstructured text corpus. Secondly, task-specific distributed sentence representations are learned by a one-dimensional CNN [8] using the previously learned word embeddings as pre-trained input vectors from a sentence level corpus. Finally, task-specific distributed paragraph representations are learned from a paragraph or document level corpus by the same CNN using the previously learned sentence vectors as pre-trained input vectors. The obtained paragraph representations can be used as the input to classification algorithms for further processing.

We evaluate our model on two tasks, topic classification on the DBpedia ontology classification dataset [2] [1] and sentiment classification on the Amazon review dataset from the Stanford Network Analysis Project (SNAP) [11] [2]. Our model achieves the state-of-the-art results, yielding a relative improvement of more than 20.61% and 19.95% in terms of error rate, respectively.

2 Our Approach

In this study, we propose to learn task-specific distributed paragraph representations by learning distributed word, sentence and paragraph representations hierarchically. The system framework with three main components is shown in Fig.1. The first component, the *Distributed Word Representation Model*, generates word embeddings from a large collection of raw text. These word embeddings are then fed into the *Distributed Sentence Representation Tier*, in which a one-dimensional CNN takes a sentence-level

[1] https://drive.google.com/folderview?id=0Bz8a_Dbh9Qh bfll6bVpmNUtUcFdjYmF2SEpmZ UZUcVNiMUw1TWN6RDV3a0JHT3kxLVhVR2M&usp=sharing

[2] https://snap.stanford.edu/data/web-Amazon.html

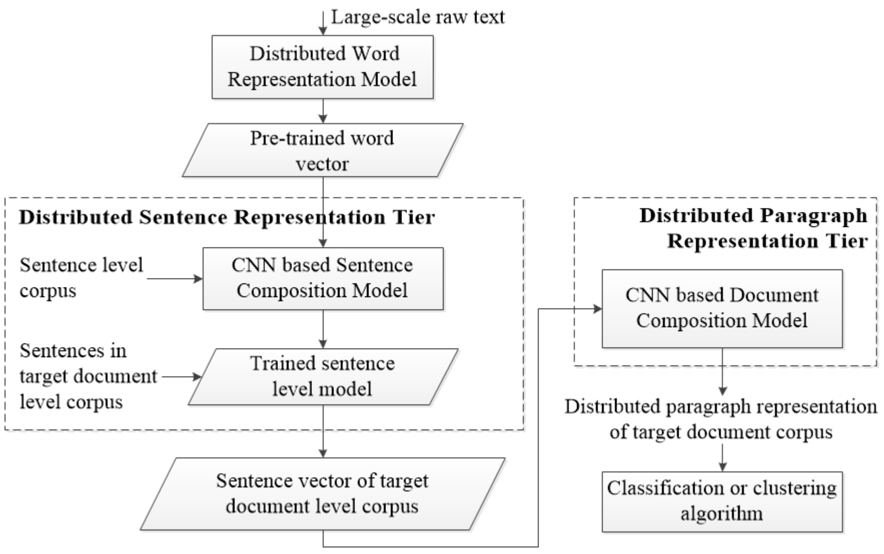
Large-scale raw text
Distributed Word Representation Model
Pre-trained word vector
Distributed Sentence Representation Tier
Sentence level corpus
CNN based Sentence Composition Model
Sentences in target document level corpus
Trained sentence level model
Sentence vector of target document level corpus
Distributed Paragraph Representation Tier
CNN based Document Composition Model
Distributed paragraph representation of target document corpus
Classification or clustering algorithm

Fig. 1. The framework of our approach.

corpus to train a sentence composition model which generates the distributed sentence representation for sentences. The third component, the *Distributed Paragraph Representation Tier* takes sentence representations as input to train distributed document representation of the target document-level corpus. We call our proposed framework two-tier convolutional neural network (2-TCNN).

2.1 Distributed Word Representation Model

The CBOW model introduced by Mikolov et al. [12] is used to learning distributed word representations. The training objective of CBOW model is to use the surrounding words of the target word in a sentence or a document to predict word representations. It can capture a large number of syntactic and semantic word relationships from unstructured text data.

Given a sequence of training words $w_1, w_2, w_3 \dots w_T$, the training objective is to maximize the average log probability:

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

where c is the size of the training context, w_t is the center word, and $\log p(w_t|w_{t+i})$ is the conditional log probability of the center word w_t given the surrounding words w_{t+i}. The prediction task is performed via softmax. The *hierarchical softmax* [15, 14] process which uses a binary tree representation of the output layer with the words as leaves, is used to reduce computational complexity.

2.2 Distributed Sentence Representation Tier

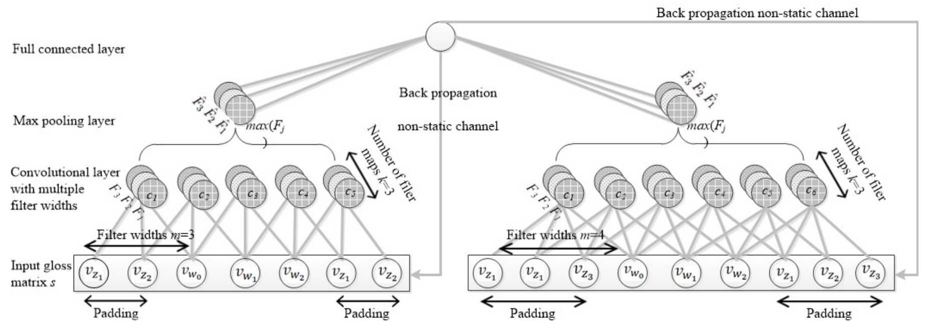
Back propagation non-static channel
Full connected layer
Back propagation
non-static channel
Max pooling layer
$\hat{F}_3\ \hat{F}_2\ \hat{F}_1$
$max(F_j)$
Number of filter maps k=3
Convolutional layer with multiple filter widths
$F_3\ F_2\ F_1$
c_1 c_2 c_3 c_4 c_5 c_6
Filter widths m=3
Filter widths m=4
Input gloss matrix s
v_{z_1} v_{z_2} v_{w_0} v_{w_1} v_{w_2} v_{z_1} v_{z_2}
v_{z_1} v_{z_1} v_{z_3} v_{w_0} v_{w_1} v_{w_2} v_{z_1} v_{z_2} v_{z_3}
Padding

Fig. 2. A one-dimensional CNN with 2 filter widths.

The one-dimensional CNN[3] proposed by Kim [8] is used to learn distributed sentence representation from a sentence-level corpus. It is a slight variant of the architecture[4] proposed by Collobert and Weston [3]. As shown in Fig.2, It takes the word embedding matrix s as an input where each column corresponds to the distributed representation $v_{w_i} \in \mathbb{R}^d$ of a word w_i in the sentence or padding vector $v_{z_i} \in \mathbb{R}^d$:

$$s = [v_{z_1}, \cdots, v_{z_{m-1}}, v_{w_1}, \cdots, v_{w_n}, v_{z_1}, \cdots, v_{z_{m-1}}], \tag{2}$$

where v_{w_i} is a d dimensional pre-trained word vector. v_{z_i} is d dimensional zero vector. m is the size of filter window. n is defined as the max length of sentences in the training set.

The idea behind the one-dimensional convolution is to take the dot product of the vector w with each m-gram in the sentence s to obtain another sequence c. In the convolutional layer, one-dimensional convolution is taken between a filter vector $w \in \mathbb{R}^{md}$ and a vector $s_{i:i+m-1} \in \mathbb{R}^{md}$ of m concatenated columns in s. The i-th feature $c_i \in \mathbb{R}$ of a feature map $F_j \in \mathbb{R}^{n+m-1}$ is generated as follows:

$$c_i = f(w \cdot s_{i:i+m-1} + b) \tag{3}$$

Where $b \in \mathbb{R}$ is a bias term and f is a point-wise non-linear function such as the hyperbolic tangent. $s_{i:i+m-1}$ refers to the i-th to $(i+m-1)$-th column of s. A feature map $F_j \in \mathbb{R}^{n+m-1}$ is defined as

$$F_j = [c_1, c_2, \cdots, c_{n+m-1}] \tag{4}$$

In the pooling layer, a max-overtime pooling operation [4], which forces the network to capture the most useful local features produced by the convolutional layers, is

[3] https://github.com/yoonkim/CNN_sentence
[4] http://ronan.collobert.com/senna/

applied over F_j. The maximum value $\hat{F_j} = max(F_j)$ taken as the feature corresponding to a particular filter w. k-$\hat{F_j}$ concatenates to a vector $\hat{F} \in \mathbb{R}^k$.

The model uses multiple filters (with varying window sizes) to obtain multiple features. These features form the penultimate layer and are passed to a fully connected softmax layer whose output is the probability distribution over labels. In non-static channel, the training error propagates back to fine-tune the parameters (w, b) and the input word vectors.

The vector generated in the penultimate layer of the CNN architecture is regarded as distributed sentence representation which captures the semantic content of the input sentence, in some degree. For training the distributed sentence representation tier, we use sentences from a sentence level corpus as our positive training examples, and randomly replace part of the sentences (for example, half of the words) to construct negative training examples. Sentences in a target document corpus are fed to the trained CNN model and the corresponding sentence vectors are output by the penultimate layer of the CNN.

2.3 Distributed Paragraph Representation Tier

The distributed paragraph representation tier of 2-TCNN has the same architecture as the sentence representation tier. The difference is that it takes sentence vectors of a document corpus as input instead of using word embeddings. The vector generated in the penultimate layer of the CNN in this tier is regarded as distributed paragraph representation which captures the semantic content of the document corpus. The paragraph vectors learned for both the training set and the test set of a document corpus can be used as features in a supervised classification algorithm.

3 Evaluation and Discussion

3.1 Experimental Settings

We evaluate our proposed framework on two supervised classification tasks, topic classification on the DBpedia ontology dataset [18] and sentiment classification of Amazon reviews.

In all experiments, we use the publicly available word embeddings trained by the CBOW model on 100 billion words from Google News [5] as pre-trained word embeddings. Words not present in the set of pre-trained words are initialized randomly in the same way as in [12].

To train the sentence representation tier of our model, we use the glosses in WordNet [13] as positive training samples for the DBpedia ontology classification experiments. There are 117,791 glosses in the WordNet 3.1 version. We randomly replace half of words in a positive sample to construct a negative training sample. In Amazon reviews sentiment analysis experiments, we use Stanford sentiment tree bank[6] [17] to train the sentence composition model.

[5] https://drive.google.com/file/d/0B7XkCwpI5KDYNlNU TTlSS21pQmM/edit?usp=sharing
[6] http://nlp.stanford.edu/sentiment/index.html

Both sentence vectors and paragraph vectors are trained using the same CNN settings. We use: rectified linear units, filter windows of 3, 4, 5 with 100 feature maps each, AdaDelta decay parameter of 0.95, dropout rate of 0.5.

For supervised classification, we use the LIBLINEAR tools [5] with default parameter settings in Weka [6].

We compare four models on distributed paragraph representations learning: word2vec, Bag of Words, ConvNet and 2-TCNN. The *word2vec* model refers to a bag-of-centroids model via word2vec[12], in which k-means algorithm are applied on word vectors learned from Google News corpus with $k = 5000$, and then a bag of these centroids are used for multinomial logistic regression. *Bag of Words* refers to the bag-of-words model, in which appearance frequency of each word in the training dataset are counted, and 5,000 most frequent ones are chose as the bag of features for multinomial logistic regression. The *ConvNet* model proposed by Zhang and LeCun [18] has four variants depending on the scale of the network and whether a thesaurus is used or not. Both the large and small ConvNet are 9 layers deep with 6 convolutional layers and 3 fully-connected layers, with different number of hidden units and frame sizes. For more details, please refer to [18]. Finally, *2-TCNN* is our proposed model.

3.2 DBpedia Ontology Classification

The DBpedia ontology classification dataset [18] is constructed by choosing 14 non-overlapping classes from DBpedia 2014 [10]. From each class, 40,000 training samples and 5,000 testing samples are selected randomly.

Table 1. Experimental results on the DBpedia ontology classification dataset.

Model	**Error rate**
word2vec	10.59%
Bag of Words	3.57%
Small ConvNet without thesaurus	2.01%
Small ConvNet with thesaurus	1.85%
Large ConvNet without thesaurus	1.74%
Large ConvNet with thesaurus	1.60%
2-TCNN	**1.27**%

In Table 1, we report the classification error rates of different models on this dataset where the results of other models have been previously reported in [18]. It is observed that the *Bag of Words* model goes beyond the barrier of 5% error rate. It is a very significant improvement compared to *word2vec*. More significant improvement is observed when distributed representation are used in the ConvNet model. *Large ConvNet with thesaurus* achieves the best performance among the baseline models. Our model improves upon the best ConvNet model by a further 0.33% on error rate, showing a relative improvement of 20.61%.

3.3 Amazon Review Sentiment Classification

The Amazon review sentiment analysis dataset from the Stanford Network Analysis Project (SNAP) contains review texts with 5 score labels from 1 to 5. In order to construct a binary sentiment classification dataset, reviews with rating 1 and 2 are treated as negative reviews, while reviews with rating 4 and 5 are taken as positive reviews. We use the same training/testing split as in [18] where the training set contains 3,000,000 samples for each positive and negative class and the test set contains 450,000 samples.

Table 2. Experimental results on the Amazon review sentiment classification dataset.

Model	**Error rate**
word2vec	16.93%
Bag of Words	14.46%
Small ConvNet without thesaurus	4.16%
Small ConvNet with thesaurus	3.99%
Large ConvNet without thesaurus	3.66%
Large ConvNet with thesaurus	3.92%
2-TCNN	**3.47**%

The error rates of different models on this dataset are shown in Table 2 where the results of other models have been previously reported in [18]. It is observed that the *Bag of Words* model performs better than *word2vec*, inline with our observation on the DBpedia ontology classification task. The ConvNet models significantly outperform these two models. It is somewhat surprising that *Large ConvNet without thesaurus* achieves better performance than *Large ConvNet with thesaurus*, showing that data augmentation techniques do not always work well. Our model outperforms the best ConvNet model by 0.19% on error rate, a relative improvement of 5.19%. It shows the effectiveness of our proposed learning model for distributed paragraph representations.

3.4 Discussion

All the baseline models take an input paragraph or a short document as a long sentence without any structures or sentence-level annotation information. By making use of knowledge about words, sentences and paragraphs, our model achieves the best performance on both the datasets.

Both the large and small ConvNet are 9 layers deep with 6 convolutional layers and 3 fully-connected layers. Our model is only 2 layers deep. It was reported in [18] that using an NVIDIA Tesla K40, training took about 5 hours per epoch for the large model, and 2 hours for the small model. Our model was trained using GeForce GTX TITAN X and only took less than 10 minutes per epoch on the DBpedia ontology classification dataset.

4 Conclusion and Future Directions

This paper presents a two-tier convolutional neural network model to learn distributed paragraph representations. For each paragraph in a corpus, the model generates the high quality distributed paragraph representations by using the text structure and task-specific annotation information. The obtained paragraph representations achieve the state-of-the-art results on the DBpedia ontology classification dataset and the Amazon review sentiment analysis dataset. The results show the effectiveness of our proposed learning framework for distributed paragraph representations. Our model is only two layers deep. It is much more efficient than the previously proposed ConvNet models with far more layers. Future work includes further improving the proposed method and applying the paragraph vectors to other NLP tasks.

Acknowledgments

This work is supported by the National Natural Science Foundation of China (No. 61370165, 61203378), National 863 Program of China 2015AA015405, the Natural Science Foundation of Guangdong Province (No. S2013010014475), Shenzhen Development and Reform Commission Grant No.[2014]1507, Shenzhen Peacock Plan Research Grant KQCX20140521144507925 and Baidu Collaborate Research Funding.

References

1. Bengio, Y., Ducharme, R., Vincent, P.: A neural probabilistic language model. Journal of Machine Learning Research 3, 1137–1155 (2003)
2. Bizer, C., Lehmann, J., Kobilarov, G., Auer, S., Becker, C., Cyganiak, R., Hellmann, S.: Dbpedia-a crystallization point for the web of data. Web Semantics: science, services and agents on the world wide web 7(3), 154–165 (2009)
3. Collobert, R., Weston, J.: A unified architecture for natural language processing: Deep neural networks with multitask learning. In: Proceedings of the 25th International Conference on Machine Learning (ICML). pp. 160–167. ACM (2008)
4. Collobert, R., Weston, J., Bottou, L., Karlen, M., Kavukcuoglu, K., Kuksa, P.: Natural language processing (almost) from scratch. The Journal of Machine Learning Research 12, 2493–2537 (2011)
5. Fan, R.E., Chang, K.W., Hsieh, C.J., Wang, X.R., Lin, C.J.: Liblinear: A library for large linear classification. The Journal of Machine Learning Research 9, 1871–1874 (2008)
6. Hall, M., Frank, E., Holmes, G., Pfahringer, B., Reutemann, P., Witten, I.H.: The weka data mining software: an update. ACM SIGKDD explorations newsletter 11(1), 10–18 (2009)
7. Kalchbrenner, N., Grefenstette, E., Blunsom, P.: A convolutional neural network for modelling sentences. In: Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics (ACL). pp. 655–665. Association for Computational Linguistics, Baltimore, Maryland (June 2014)
8. Kim, Y.: Convolutional neural networks for sentence classification. In: Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP). pp. 1746–1751 (October 2014)

9. Le, Q.V., Mikolov, T.: Distributed representations of sentences and documents. In: Proceedings of the 31th International Conference on Machine Learning (ICML). pp. 1188–1196 (2014)
10. Lehmann, J., Isele, R., Jakob, M., Jentzsch, A., Kontokostas, D., Mendes, P.N., Hellmann, S., Morsey, M., van Kleef, P., Auer, S., et al.: Dbpedia-a large-scale, multilingual knowledge base extracted from wikipedia. Semantic Web Journal 5, 1–29 (2014)
11. McAuley, J., Leskovec, J.: Hidden factors and hidden topics: understanding rating dimensions with review text. RecSys (2013)
12. Mikolov, T., Chen, K., Corrado, G., Dean, J.: Efficient estimation of word representations in vector space. In: Proceedings of Workshop at the International Conference on Learning Representations (ICLR) (2013)
13. Miller, G.A.: Wordnet: a lexical database for english. Communications of the ACM 38(11), 39–41 (1995)
14. Mnih, A., Hinton, G.E.: A scalable hierarchical distributed language model. In: Advances in neural information processing systems. pp. 1081–1088 (2009)
15. Morin, F., Bengio, Y.: Hierarchical probabilistic neural network language model. In: Proceedings of the international workshop on artificial intelligence and statistics. pp. 246–252. Citeseer (2005)
16. dos Santos, C.N., Gatti, M.: Deep convolutional neural networks for sentiment analysis of short texts. In: Proceedings of the 25th International Conference on Computational Linguistics (COLING), Dublin, Ireland (2014)
17. Socher, R., Perelygin, A., Wu, J.Y., Chuang, J., Manning, C.D., Ng, A.Y., Potts, C.: Recursive deep models for semantic compositionality over a sentiment treebank. In: Proceedings of the Conference on Empirical Methods in Natural Language Processing (EMNLP). pp. 1631–1642. Citeseer (2013)
18. Zhang, X., LeCun, Y.: Text understanding from scratch. arXiv preprint arXiv:1502.01710 (2015)