Learning Distributed Word Representations For Bidirectional LSTM Recurrent Neural Network*

Peilu Wang[1,2]†, **Yao Qian**[3], **Hai Zhao**[1]‡, **Frank K. Soong**[2], **Lei He**[2], **Ke Wu**[1]
[1]Shanghai Jiao Tong University, Shanghai, China
[2]Microsoft Research Asia, Beijing, China
[3]Educational Testing Service Research, CA, USA
peiluwang@163.com, yqian@ets.org, zhaohai@cs.sjtu.edu.cn
{frankkps, helei}@microsoft.com,wuke@cs.sjtu.edu.cn

Abstract

Bidirectional long short-term memory (BLSTM) recurrent neural network (RNN) has been successfully applied in many tagging tasks. BLSTM-RNN relies on the distributed representation of words, which implies that the former can be futhermore improved through learning the latter better. In this work, we propose a novel approach to learn distributed word representations by training BLSTM-RNN on a specially designed task which only relies on unlabeled data. Our experimental results show that the proposed approach learns useful distributed word representations, as the trained representations significantly elevate the performance of BLSTM-RNN on three tagging tasks: part-of-speech tagging, chunking and named entity recognition, surpassing word representations trained by other published methods.

1 Introduction

Distributed word representations represent word with a real valued vector, which is also referred to as word embedding. Well learned distributed word representations have been shown capable of capturing semantic and syntactic regularities (Pennington et al., 2014a; Mikolov et al., 2013c) and enhancing neural network model by being used as features (Collobert and Weston, 2008; Bengio and Heigold, 2014; Wang et al., 2015).

Sequence tagging is a basic structure learning task for natural language processing. Many primary processing tasks over sentence such as word segmentation, named entity recognition and part-of-speech tagging can be formalized as a tagging task (Zhao et al., 2006; Huang and Zhao, 2007; Zhao and Kit, 2008b; Zhao and Kit, 2008a; Zhao et al., 2010; Zhao and Kit, 2011). Recently, many state-of-the-art systems of tagging related tasks are implemented with bidirectional long short-term memory (BLSTM) recurrent neural network (RNN), for example, slot filling (Mesnil et al., 2013), part-of-speech tagging (Huang et al., 2015), and dependency parsing (Dyer et al., 2015) etc. All of these systems use distributed representation of words to involve word level information. Better trained word representations would further improve the state-of-the-art performance of these tasks which makes it worthy to research the training methods of word representations.

The existing training methods of word representation can generally be divided into two classes: 1) Matrix factorization methods. These methods utilize low-rank approximation to decompose a large matrix that contains corpus statistics. One typical work is the latent semantic analysis (LSA) (Deerwester et al., 1990) in which the matrix records "term-document" information, i.e., the rows cor-

*The work was partially supported by the National Natural Science Foundation of China (Grant No. 61170114, and Grant No. 61272248), the National Basic Research Program of China (Grant No. 2013CB329401), the Science and Technology Commission of Shanghai Municipality (Grant No. 13511500200), the European Union Seventh Framework Program (Grant No. 247619), the Cai Yuanpei Program (CSC fund 201304490199, 201304490171), and the art and science interdisciplinary funds of Shanghai Jiao Tong University, No. 14X190040031, and the Key Project of National Society Science Foundation of China, No. 15-ZDA041.

†Work performed as an intern in speech group, Microsoft Research Asia

‡Corresponding author

Proceedings of NAACL-HLT 2016, pages 527–533,
San Diego, California, June 12-17, 2016.

respond to words, and the columns correspond to different documents in the corpus. Another work is hyperspace analogue to language (HAL) (Lund and Burgess, 1996) which decomposes the matrix recording "term-term" information, i.e., the rows correspond to words and columns correspond to the number of times that a word occurs in the given context. 2) Window-based methods. This type of methods learn representations by training a neural network model to make prediction within local context windows. For example, (Bengio et al., 2003) learns word representation through a feedforward neural network language model which predicts a word given its previous several words. (Collobert et al., 2011) trains a neural network to judge the validity of a given context. (Mikolov et al., 2013a) proposes skip-gram and continuous bag-of-words (CBOW) models based on a single-layer network architecture. The objective of skip-gram model is to predict the context given the word itself, while the objective of CBOW is to predict a word given its context. Aside from these two sets of methods, distributed representation can also be obtained by training recurrent neural network (RNN) language model proposed by (Mikolov et al., 2010) or GloVe model proposed by (Pennington et al., 2014a) which trains a log-bilinear model on word-word co-occurrence counts.

All of these methods suffer from shortcomings that might limit the quality of trained word distributions. The matrix factorization family only uses the statistics of co-occurrence counts, disregarding of the position of word in sentence and word order. The window-based methods only consider local context, which is incapable of involving information outside the context window. While RNN language model theoretically considers all information of the previous sequence, but fails to involve the information of the posterior sequence. The word-word co-occurence counts that GloVe model relies on also only include information within a limited sized context window.

In this paper, we propose a novel method to obtain word representation by training BLSTM-RNN model on a specifically designed tagging task. Since BLSTM-RNN theoretically involves all information of input sentence, our approach avoids those shortages suffered by most current methods.

We firstly introduce the structure of BLSTM-RNN used to learn word representations in section 2. Then the tagging task for training BLSTM-RNN is described in section 3. Experiments are presented in section 4, followed by conclusion.

2 Model Structure

The structure of BLSTM-RNN to train word representation is illustrated in Figure 1. Each input is composed of a word identity x_i^1 and additional real-valued features x_i^2. x_i^1 is represented with one-hot representation which is a binary vector with dimension $|V|$ where V is the vocabulary. The input vector I_i of the network is computed as:

$$I_i = W_1 x_i^1 + W_2 x_i^2$$

where W_1 and W_2 are weight matrixes connecting two layers and are updated with the neural network during training. $W_1 x_i^1$ is also known as the distributed representation of word or word embedding which is a real-valued vector usually with a much smaller dimension than x_i^1. Distributed representation trained in other tasks can be easily incorporated by initializing W_1 with these external representations.

3 Learning Representation

According to the structure shown in Figure 1, W_1 is a matrix of weights that is updated during training, thus the distributed representations contained in W_1 are learned simultaneously with the training of BLSTM-RNN on any supervised learning tasks. However, all such tasks for BLSTM-RNN, to the best of our knowledge, require labeled data which is usually too small in size and hard to obtain. In this section, we propose a tagging task specially for BLSTM-RNN to train distributed representations with unlabeled data.

In this method, BLSTM-RNN is applied to perform a tagging task with only two types of tags to predict: incorrect/correct. The input is a sequence of words which is a normal sentence with several words replaced by words randomly chosen from vocabulary. The words to be replaced are chosen randomly from the sentence. In practice, we generate a random number for each word, and a word is chosen to be replaced if the number is lower than a given

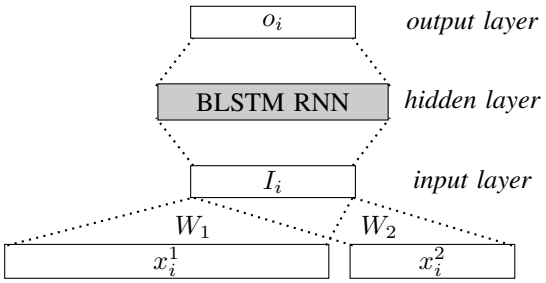

Figure 1: BLSTM-RNN for training word representation.

threshold. For those replaced words, their tags are 0 (incorrect) and for those that are not replaced, their tags are 1 (correct). A simple sample of constructed corpus is shown in Figure 2. Although it is possible that some replaced words are also reasonable in the sentence, they are still considered "incorrect". Then BLSTM-RNN is trained to judge which words have been replaced by minimizing the binary classification error on the training corpus. When the network is trained, W_1 contains all trained word representations. In our experiments, to reduce the vocabulary V, each letter of input word is transferred to its lowercase. The upper case information is kept in an additional features x_i^2 which in practice is a three-dimensional binary vector to indicate if x_i^1 is full lowercase, full uppercase or leading with a capital letter.

Our approach is similar to (Collobert and Weston, 2008) and (Gutmann and Hyvärinen, 2012; Mnih and Teh, 2012; Vaswani et al., 2013). All of these works introduce randomly sampled words and train a neural network on a binary classification task, while (Collobert and Weston, 2008) learns representations for a feedforward network and (Gutmann and Hyvärinen, 2012; Mnih and Teh, 2012; Vaswani et al., 2013) learns normalization parameters instead of representations.

4 Experiments

4.1 Experimental setup

To construct corpus for training word representations, we use North American news (Graff, 2008) which contains about 536 million words as unlabeled data. The North American news data is first tokenized with the Penn Treebank tokenizer script [1]. Consecutive digits occurring within a word are replaced with the symbol "*#*" . For example, both words "*tel92*" and "*tel6*" are converted into "*tel#*". The vocabulary is limited to the most frequent 100,000 words in North American news corpus (Graff, 2008), plus one single "UNK" symbol for replacing all out of vocabulary words. The threshold to determine whether a word is replaced is 0.2, which means about 20% tokens in corpus are replaced with tokens randomly selected from vocabulary. BLSTM-RNN is implemented based on CURRENNT (Weninger et al., 2014), an open source GPU-based toolkit of BLSTM-RNN. The dimension of word representation as well as input layer size of BLSTM-RNN is 100 and hidden layer size is 128.

Three published methods for training word representations are compared: *Skip-gram* (Mikolov et al., 2013a), *CBOW* (Mikolov et al., 2013a) and *GloVe* (Pennington et al., 2014a). They are reported superior in capturing meaningful latent structures than other previous works in (Mikolov et al., 2013a; Pennington et al., 2014a), thus are regarded as the state-of-the-art approach of training word representations. We train the *Skip-gram* and *CBOW* model using the word2vec tool (Mikolov et al., 2013b) with a context window size of 10 and 10 negative samples. For training *GloVe*, we use the GloVe tool (Pennington et al., 2014b) with a context window size 10. These configurations are set by following (Pennington et al., 2014a). Training corpus, vocabulary and dimension of word representations are set the same as that in experiment for training word representations with BLSTM-RNN[2].

[1] `https://www.cis.upenn.edu/~treebank/tokenization.html`

[2] Our experimental setup are released at: `https://github.com/PeiluWang/naacl2016_blstmwe`

Original Sentence:
They seem to be prepared to make $\cdots$

Input Sentence:
They <u>*beast*</u> *to be* <u>*austere*</u> *to make* $\cdots$

Tag Sequence:
1 0 1 1 0 1 1 $\cdots$

Figure 2: Sample of constructed corpus for training word representations. Two words "*seem*" and "*prepared*" are replaced with words randomly chosen from vocabulary.

Sys	**POS**(Acc.)	**CHUNK**(F1)	**NER**(F1)
BLSTM	96.60	91.71	82.52
BLSTM+CBOW	96.73	92.14	84.37
BLSTM+Skip	96.85	92.45	85.80
BLSTM+GloVe	97.02	93.01	87.33
BLSTM+BLSTMWE	**97.26**	**94.44**	**88.38**

Table 1: Performance of BLSTM-RNN with different representations on three tagging tasks

4.2 Evaluation

The quality of trained distributed representation is evaluated by the performance of BLSTM-RNN which uses the trained representations on practical tasks. The representations which lead to better performance are considered containing more useful latent information and are judged better. The structure of BLSTM-RNN to test word representations is the same as that in Figure1. To use trained representation, we initialize the weight matrix W_1 with these external representations. For words without corresponding external representations, their representations are initialized with uniformly distributed random values, ranging from -0.1 to 0.1. Three typical tagging tasks are used for the evaluation: part-of-speech tagging (POS), chunking (CHUNK) and named entity recognition (NER).

- The POS tagging experiment is conducted on the Wall Street Journal data from Penn Treebank III (Marcus et al., 1993). Training, development and test sets are split according to in (Collins, 2002). Performance is evaluated by the accuracy of predicted tags on test set.

- CHUNK experiment is conducted on the data of CoNLL-2000 shared task (Sang and Buchholz, 2000). Performance is assessed by the F1 score computed by the evaluation script released by the CoNLL-2000 shared task[3].

- NER experiment is conducted on the CoNLL-2003 shared task (Tjong Kim Sang and De Meulder, 2003). Performance is measured by the F1 score calculated by the evaluation script of the CoNLL-2003 shared task [4].

To focus on the effect of word representation, for all tasks, we use the network with the same hidden structure and input features. The size of input layer is 100, size of BLSTM hidden layer is 128 and output layer size is set as the number of tag types according to the specific tagging task. Input features are composed of word identity and three-dimensional binary vector to indicate if the word is full lowercase, full uppercase or leading with a capital letter.

Table 1 presents the performance of BLSTM-RNN with different distributed representations on these three tasks. *BLSTM* is the baseline system that does not involve external word representations. Among all representations, *BLSTMWE* which is trained by our approach gets the best performance on all three tasks. It shows our approach is more helpful for BLSTM-RNN. Besides, all of the three published word representations also significantly en-

[3] http://www.cnts.ua.ac.be/conll2000/chunking

[4] http://www.cnts.ua.ac.be/conll2003/ner/

Sys	**POS**(Acc.)	**CHUNK**(F1)	**NER**(F1)
BLSTMWE (10M)	96.61	91.91	84.66
BLSTMWE (100M)	97.10	93.86	86.47
BLSTMWE (536M)	97.26	94.44	88.38

Table 2: Performance of BLSTMWE trained on corpora with different size

	Skip-gram	*CBOW*	*GloVe*	*BLSTM RNN*
Time (min.)	344	117	127	1393

Table 3: Running time of different training methods

hance BLSTM RNN. It confirms the commonly accepted notion that word representation is a useful feature.

4.3 Analysis

Table 2 shows the performance of word representations trained on corpora with different size. *BLSTMWE* (10M), *BLSTMWE* (100M) and *BLSTMWE* (536M) are word representations respectively trained by BLSTM-RNN on the first 10 million words, first 100 million words and all 536 million words of the North American news corpus. As expected, there is a monotonic increase in performance as the corpus size increases. This observation suggests that the result might be further improved by using even bigger unlabeled data.

Table 3 presents running time with different methods to train word representations on 536 million words corpus. BLSTM-RNN is trained on one NVIDIA Tesla M2090 GPU. The other three methods are trained on a 12 core, 2.53GHz Intel Xeon E5649 machine, using 12 threads. Though with the help of GPU, BLSTM-RNN is still slower than the other methods. However, it should be noted that the speed of our approach is acceptable compared with previous neural network language model based methods, including (Bengio et al., 2003; Mikolov et al., 2010; Mnih and Hinton, 2007), as our model uses a much simpler output layer which only has two nodes, avoiding the time consuming computation of the big softmax output layer in language model.

5 CONCLUSION

In this paper, we propose a novel approach to learn distributed word representations with BLSTM-RNN. Word representations are implemented as the layer weights and are obtained as a byproduct of training BLSTM-RNN on a specially designed task, thus theoretically involve information of the whole sentence. The quality of word representations are evaluated by the performance of BLSTM-RNN which uses these representations on three tagging tasks: part-of-speech tagging, chunking and named entity recognition. In experiments, word representations trained by our approach outperform the word representations trained by other published methods. Our work demonstrates an alternative way to improve BLSTM-RNN's performance by learning useful word representations.

References

Samy Bengio and Georg Heigold. 2014. Word embeddings for speech recognition. In *INTERSPEECH*, pages 1053–1057.

Yoshua Bengio, Rejean Ducharme, Pascal Vincent, and Christian Jauvin. 2003. A neural probabilistic language model. In *Journal of Machine Learning Research (JMLR)*, volume 3, pages 1137–1155.

Michael Collins. 2002. Discriminative training methods for Hidden Markov Models: Theory and experiments with perceptron algorithms. In *Proceedings of Empirical Methods in Natural Language Processing (EMNLP)*, pages 1–8.

Ronan Collobert and Jason Weston. 2008. A unified architecture for natural language processing: Deep neural networks with multitask learning. In *Proceedings of International Conference on Machine Learning (ICML)*, pages 160–167.

Ronan Collobert, Jason Weston, Léon Bottou, Michael Karlen, Koray Kavukcuoglu, and Pavel Kuksa. 2011. Natural language processing (almost) from scratch.

Journal of Machine Learning Research (JMLR), 12:2493–2537.

Scott Deerwester, Susan Dumais, Thomas Landauer, George Furnas, and Richard Harshman. 1990. Indexing by latent semantic analysis. *JASIS*, 41(6):391–407.

Chris Dyer, Miguel Ballesteros, Wang Ling, Austin Matthews, and Noah A. Smith. 2015. Transition-based dependency parsing with stack long short-term memory. In *Proceedings of Association for Computational Linguistics (ACL)*, pages 334–343.

David Graff. 2008. North American News Text, Complete LDC2008T15. `https://catalog.ldc.upenn.edu/LDC2008T15`.

Michael Gutmann and Aapo Hyvärinen. 2012. Noise-contrastive estimation of unnormalized statistical models, with applications to natural image statistics. *Journal of Machine Learning Research (JMLR)*, 13:307–361.

Changning Huang and Hai Zhao. 2007. Chinese word segmentation: A decade review. *Journal of Chinese Information Processing*, 21(3):8–20.

Zhiheng Huang, Wei Xu, and Kai Yu. 2015. Bidirectional LSTM-CRF models for sequence tagging. *arXiv:1508.01991*.

Kevin Lund and Curt Burgess. 1996. Producing high-dimensional semantic spaces from lexical co-occurrence. *Behavior Research Methods, Instruments, & Computers*, 28(2):203–208.

Mitchell Marcus, Mary Ann Marcinkiewicz, and Beatrice Santorini. 1993. Building a large annotated corpus of English: The Penn Treebank. *Computational linguistics (CL)*, 19(2):313–330.

Grégoire Mesnil, Xiaodong He, Li Deng, and Yoshua Bengio. 2013. Investigation of recurrent-neural-network architectures and learning methods for spoken language understanding. In *INTERSPEECH*, pages 3771–3775.

Tomas Mikolov, Martin Karafiát, Lukas Burget, Jan Cernockỳ, and Sanjeev Khudanpur. 2010. Recurrent neural network based language model. In *INTERSPEECH*, pages 1045–1048.

Tomas Mikolov, Kai Chen, Greg Corrado, and Jeffrey Dean. 2013a. Efficient estimation of word representations in vector space. *arXiv preprint arXiv:1301.3781*.

Tomas Mikolov, Kai Chen, Greg Corrado, and Jeffrey Dean. 2013b. word2vec. `https://code.google.com/p/word2vec/`.

Tomas Mikolov, Ilya Sutskever, Kai Chen, Greg Corrado, and Jeff Dean. 2013c. Distributed representations of words and phrases and their compositionality. In *Proceedings of Conference on Neural Information Processing Systems (NIPS)*, pages 3111–3119.

Andriy Mnih and Geoffrey Hinton. 2007. Three new graphical models for statistical language modelling. In *Proceedings of International Conference on Machine Learning (ICML)*, pages 641–648.

Andriy Mnih and Yee Whye Teh. 2012. A fast and simple algorithm for training neural probabilistic language models. In *Proceedings of International Conference on Machine Learning (ICML)*, pages 1751–1758.

Jeffrey Pennington, Richard Socher, and Christopher Manning. 2014a. Glove: Global vectors for word representation. In *Proceedings of Empirical Methods in Natural Language Processing (EMNLP)*, pages 1532–1543.

Jeffrey Pennington, Richard Socher, and Christopher D. Manning. 2014b. GloVe. `http://nlp.stanford.edu/projects/glove/`.

Erik Tjong Kim Sang and Sabine Buchholz. 2000. Introduction to the CoNLL-2000 shared task: Chunking. In *Proceedings of Conference on Natural Language Learning (CoNLL)*, pages 127–132.

Erik Tjong Kim Sang and Fien De Meulder. 2003. Introduction to the CoNLL-2003 shared task: Language-independent named entity recognition. In *Proceedings of Conference on Natural Language Learning (CoNLL)*, pages 142–147.

Ashish Vaswani, Yinggong Zhao, Victoria Fossum, and David Chiang. 2013. Decoding with large-scale neural language models improves translation. In *Proceedings of Empirical Methods in Natural Language Processing (EMNLP)*, pages 1387–1392.

Peilu Wang, Yao Qian, Frank Soong, Lei He, and Hai Zhao. 2015. Word embedding for recurrent neural network based tts synthesis. In *Proceedings of International Conference on Acoustics, Speech, and Signal Processing (ICASSP)*, pages 4879–4883.

Felix Weninger, Johannes Bergmann, and Björn Schuller. 2014. Introducing CURRENNT–the Munich open-source CUDA recurrent Neural Network Toolkit. *Journal of Machine Learning Research (JMLR)*, 16(1):547–551.

Hai Zhao and Chunyu Kit. 2008a. An empirical comparison of goodness measures for unsupervised chinese word segmentation with a unified framework. In *Proceedings of International Joint Conference on Natural Language Processing (IJCNLP)*, volume 1, pages 9–16.

Hai Zhao and Chunyu Kit. 2008b. Unsupervised segmentation helps supervised learning of character tagging for word segmentation and named entity recognition. In *SIGHAN Workshop on Chinese Language Processing*, pages 106–111.

Hai Zhao and Chunyu Kit. 2011. Integrating unsupervised and supervised word segmentation: The

role of goodness measures. *Information Sciences*, 181(1):163–183.

Hai Zhao, Chang-Ning Huang, and Mu Li. 2006. An improved chinese word segmentation system with conditional random field. In *SIGHAN Workshop on Chinese Language Processing*, pages 162–165.

Hai Zhao, Chang-Ning Huang, Mu Li, and Bao-Liang Lu. 2010. A unified character-based tagging framework for chinese word segmentation. *ACM Transactions on Asian Language Information Processing (TALIP)*, 9(2):1–32.