Compound Embedding Features for Semi-supervised Learning

Mo Yu[1], Tiejun Zhao[1], Daxiang Dong[2], Hao Tian[2] and Dianhai Yu[2]
Harbin Institute of Technology, Harbin, China
Baidu Inc., Beijing, China
{yumo,tjzhao}@mtlab.hit.edu.cn
{dongdaxiang,tianhao,yudianhai}@baidu.com

Abstract

To solve data sparsity problem, recently there has been a trend in discriminative methods of NLP to use representations of lexical items learned from unlabeled data as features. In this paper, we investigated the usage of word representations learned by neural language models, i.e. word embeddings. The direct usage has disadvantages such as large amount of computation, inadequacy with dealing word ambiguity and rare-words, and the problem of linear non-separability. To overcome these problems, we instead built compound features from continuous word embeddings based on clustering. Experiments showed that the compound features not only improved the performances on several NLP tasks, but also ran faster, suggesting the potential of embeddings.

1 Introduction

Supervised learning methods have achieved great successes in the field of ***Natural Language Processing (NLP)***. However, in practice most methods are usually limited by the problem of data sparsity, since it is impossible to obtain sufficient labeled data for all NLP tasks. In these situations semi-supervised learning can help to make use of both labeled data and easy-to-obtain unlabeled data.

The semi-supervised framework that is widely applied to NLP is to first learn word representations, which are feature vectors of lexical items, from unlabeled data and then plug them into a supervised system. These methods are very effective in utilizing large-scale unlabeled data and have successfully improved performances of state-of-the-art supervised systems on a variety of tasks (Koo et al., 2008; Huang and Yates, 2009; Täckström et al., 2012).

With the development of ***neural language models (NLM)*** (Bengio et al., 2003; Mnih and Hinton, 2009), recently researchers become interested in word representations (also called **word embeddings**) learned by these models. Word embeddings are dense low dimensional real-valued vectors. They are composed of some latent features, which are expected to capture useful syntactic and semantic properties. Word embeddings are usually served as the first layer in deep learning systems for NLP (Collobert and Weston, 2008; Socher et al., 2011a, 2011b) and help these systems perform comparably with the state-of-the-art models based on handcrafted features. They also have been directly added as features to the state-of-the-art models of chunking and NER, and have achieved significant improvements (Turian et al. 2010).

Although the direct usage of continuous embeddings has been proved to be an effective method for enhancing the state-of-the-art supervised models, it has some disadvantages, which made them be out-performed by simpler Brown cluster features (Turian et al, 2010) and made them computationally complicated. Firstly, embeddings of rare words are insufficiently trained since they are only updated few times and are close to their random initial values. As shown in (Turian et al, 2010), this is the main reason that models with embedding features made more errors than those with Brown cluster features. Secondly, in NLMs, each word has its unique representation, so it is difficult to represent different senses for ambiguous words. Thirdly, word embeddings are unsuitable for linear models in some tasks as will be proved in Section

Proceedings of NAACL-HLT 2013, pages 563–568,
Atlanta, Georgia, 9–14 June 2013.

4.2. This is possibly because in these tasks, either the target labels are correlated with combinations of different dimensions of word embeddings, or discriminative information may be coded in different intervals in the same dimension. So treating embeddings directly as inputs to a linear model could not fully utilize them. Moreover, since embeddings are dense vectors, it will introduce large amount of computations when they are directly used as inputs, making the method impractical.

In this paper, we first introduced the idea of clustering embeddings to overcome the last two disadvantages discussed above. The high-dimensional cluster features make samples from different classes better separated by linear models. And models with these features can still run fast because the clusters are sparse and discrete.

Second, we proposed the compound features based on clustering. Compound features, which are conjunctive features of neighboring words, have been widely used in NLP models for improving the performances because they are more discriminative. Compound features of embeddings can also help a model to better predict labels associated with rare-words and ambiguous words, because compound features composed of embeddings of nearby words can help to better describe the property of these words. Compound features are difficult to build on dense embeddings. However they are easy to induce from the sparse embedding clusters proposed in this paper.

Experiments on chunking and NER showed that based on the same embeddings, the compound features managed to achieve better performances. Moreover, we proposed analyses to reveal the reasons for the improvements of embedding-clusters and compound features. They suggest that these features can better deal with rare-words and word ambiguity, and are more suitable for linear models.

In addition, although Brown clustering was considered better in (Turian et al 2010), our experiment results and comparisons showed that our compound features from embedding clustering is at least comparable with those from Brown clustering. Since embeddings can greatly benefit from the improvement and developing of deep learning in the future, we believe that our proposed method has a large space of performance growth and will benefit more applications in NLP.

In the rest of the paper, Section 2 introduces how compound embedding features were obtained. Section 3 gives experimental results. In Section 4, we give analysis about the advantages of compound features. Section 5 gives the conclusions.

2 Clustering of Word Embeddings

2.1 Learning Word Embeddings

Word embeddings in this paper were trained by NLMs (Bengio et al., 2003). The model predicts the scores of probabilities of words given their context information in the sentences. It first converts the current word and its context words (e.g. n-1 words before it as in n-gram models) into embeddings. Then these embeddings are put together and propagate forward on the network to compute the score of current word. After minimizing the loss on training data, embeddings are learned and can be further used as smoothing representations for words.

2.2 Clustering of embeddings

In order to get compound features of embeddings, we first induce discrete clusters from the embeddings. Concretely, the *k*-means clustering algorithm is used. Each word is treated as a single sample. A cluster is represented as the mean of the embeddings of words assigned to it. Similarities between words and clusters are measured by Euclidean distance. As discussed and experimented later, different numbers of *k*s contain information of different granularity. So we combine clustering results achieved by different *k*s as features to better utilize the embeddings.

2.3 Compound features

Based on embedding clusters, more powerful compound features can be built. Compound features are conjunctions between basic features of words and their contexts, which are widely used in NLP. Koo et al. (2008) also observed that compound features of Brown clusters achieved more improvements on parsing.

It is also necessary to build compound embedding features since they can better deal with rare-words and ambiguous words. For example, although embedding of a rare-word is not fully trained and hence inaccurate, embeddings of its context words can still be accurate as long as they are not rare and are fully trained. So we could utilize the combination of embeddings before and after the word to predict its tag correctly. We conducted analysis to verify our theory in Section4.

We combined the compound features together with other state-of-the-art human-craft features in supervised models. Examples of the resulted feature templates in chunking and NER are shown in Table 1 & 2. The feature $y_{-1}/y_0/c_{-1}/c_1$ in the last row is an example of compound feature made up of the embedding clusters of words before and after current word. Compound feature extraction can similarly be applied to form compound features of Brown clusters. For example, Brown clusters can replace embedding clusters in 3th row of Table 1.

Words	$w_{i,i\in\{-2:2\}}, w_{i-1}/w_{i,i\in\{0,1\}}$
POS	$p_{i,i\in\{-2:2\}}, p_{i-1}/p_{i,i\in\{-1,2\}}$
Cluster	$c_{i,i\in\{-2:2\}}, c_{i-1}/c_{i,i\in\{0,1\}}, c_{-1}/c_1$
Transition	$y_{-1}/y_0/\{w_0, p_0, c_0, c_{-1}/c_1\}$

Table 1: Chunking features. Cluster features are suitable for both Brown clusters and embedding clusters. Symbol y_i is the tag predicted on word w_i.

Words	$w_{i,i\in\{-2:2\}}, w_{i-1}/w_{i,i\in\{0,1\}}$
Pre/suffix	$w^{1:i}_{0,i\in\{2:4\}}, w^{-i:-1}_{0,i\in\{1:4\}}$
Orthography	$Hyp(w_0), Cap(w_0)$
POS	$p_{i,i\in\{-2:2\}}, p_{i-1}/p_{i,i\in\{-1,2\}}$
Chunking	$b_{i,i\in\{-2:2\}}, b_{i-1}/b_{i,i\in\{-1,2\}}$
Cluster	$c_{i,i\in\{-2:2\}}, c_{i-1}/c_{i,i\in\{0,1\}}, c_{-1}/c_1$
Transition	$y_{-1}/y_0/\{w_0, p_0, c_0, c_{-1}/c_1\}$

Table 2: NER features. Hyp indicates if word contains hyphen and Cap indicates if first letter is capitalized.

3 Experiments

3.1 Experimental settings

We tested our compound features on the same chunking (CoNLL2000) and NER (CoNLL2003) tasks in (Turian et al., 2010). The Brown cluster features were used for comparison, which shared the same feature template used by clusters of embeddings. To compare with the work of (Turian et al, 2010), which aimed to solve the same problem but using embedding directly, we used the same word embeddings (CW 50) and Brown clusters (1000 clusters) they provided. The embeddings in (Turian et al, 2010) are trained on RCV corpus, while the CoNLL2000 data is a part of the WSJ corpus. Since we believe that word representations trained on similar domain may better help to improve the results, we also used embeddings and Brown clusters trained on unlabeled WSJ data from (Nivre et al, 2007) for comparison.

Moreover, we wish to find out whether our method extends well to languages other than English. So we conducted experiments on Chinese NER, where large amount of training data exists, which makes improving accuracies more difficult. We used data from People's Daily (Jan.-Jun. 1998) and converted them following the style of Penn CTB (Xue et al, 2005). Data from April was chosen as test set (1,309,616 words in 55,177 sentences), others for training (6,119,063 words in 255,951 sentences). The Chinese word representations were trained on Chinese Wikipedia until March 2011. The features used in Chinese NER are similar to those in English, except for the orthography, pre/suffixes, and chunking features.

We did little pre-processing work for the training of word representations on WSJ data. The datasets were tokenized and capital words were kept. For training of Chinese Wikipedia, we retained the bodies of all articles and replaced words with frequencies lower than 10 as an "UK_WORD" token. On each dataset, we induced embeddings with 64 dimensions based on 7-gram models and 1000 Brown clusters. The method in (Schwenk, 2007) was used to accelerate the training processes of NLMs. All the NLMs were trained for 5 epochs.

For clustering of embeddings we choose k=500 and 2500 since such combination performed best on development set as shown in the next section. We chose the Sofia-ml toolkit (Sculley 2010) for clustering of embeddings in order to save time.

In the experiments CRF models were used and were optimized by ASGD (implemented by Léon Bottou). For comparison we re-implemented the direct usage of embeddings in (Turian et al, 2010) with CRFsuite (Okazaki, 2007) since their features contain continuous values.

3.2 Performances

Table 3 shows the chunking results. The results reported in (Turian et al. 2010) were denoted as "**direct**". Based on the same word representations, our compound features got better performances in all cases. The embedding features trained on unlabeled WSJ data yield further improvements, showing that word representations from similar domains can better help the supervised tasks.

System	Direct	Compound
Baseline	93.75	
+Embedding (RCV)	94.10	94.19
+Brown (RCV)	94.11	94.24
+Brown&Emb (RCV)	94.35	**94.42**
+Embedding (WSJ)	94.20	94.37
+Brown (WSJ)	94.25	94.36
+Brown&Emb (WSJ)	94.43	**94.58**

Table 3: F1-scores of chunking

In the experiments of NER, first we evaluated how the numbers of clusters *k* will affect the performances on development set (Figure 1). The results showed that both the cluster features (excluding all compound embedding features) and compound features could achieve better results than direct usage of the same embeddings. It also showed that the performances did not vary much when *k* was between 500 and 3000. When *k*=2500, the result was a little higher than others. We finally chose combination of *k*=500 and 2500, which achieved best results on development set.

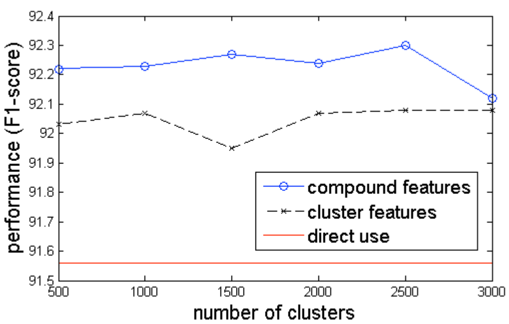

Figure 1: Relation between numbers of clusters k and performances on development set.

The performances of NER on test set are shown in Table 4. Our baseline is slightly lower than that in (Turian et al, 2010), because the first-order CRF cannot utilize context information of NE tags. Despite of this, same conclusions with chunking held.

System	Direct	Compound
Baseline	83.78	
+Embedding	87.38	88.46
+Brown	88.14	88.23
+Brown&Embedding	88.85	89.06

Table 4: F1-scores of English NER on test data

Performances on Chinese NER are shown in Table 5. Similar results were observed as in English NER, showing that our method extends to other languages as well.

System	Direct	Compound
Baseline	88.24	
+Embedding	89.98	90.37
+Brown	90.24	90.55
+Brown&Embedding	90.66	**90.96**

Table 5: F1-scores of Chinese NER on test data

Above results gave evidences that although clustering embeddings may lose some information, the derived compound features did have better performances. The compound features can also improve the performances of Brown clusters, but not as much as they did on embeddings. And the combination of embedding-clusters and Brown-clusters could further improve the performances, since they made use of different type of context information.

The compound features also reduced the time cost of using embedding features. For example, the time for tagging one sentence in English NER was reduced from 5.6 ms to 1.6 ms, shown in Table 6.

Embedding	Time (ms)
Baseline	1.2
Embeddings (direct)	5.6
Embeddings (compound)	1.6

Table 6: Running time of different features

4 Analysis

Our compound embedding features greatly outperformed the direct usage of same embeddings on English NER. In this section we conducted analyses to show the reasons for the improvements.

4.1 Rare-words and ambiguous words

To show the compound features have stronger abilities to handle rare words, we counted the numbers of errors made on words with different frequencies on unlabeled data. Here the word frequencies are from the results of Brown clustering provided by (Turian et al. 2010). We compared our compound embedding features with direct usage of embeddings as well as Brown clusters, which is believed to work better on rare words. Figure 2(a) shows that the compound features indeed resulted in fewer errors than the two baseline methods in most cases. Errors of embeddings occurred on words with frequencies lower than 2K and those in the range of 16 to 256 were reduced by 10.55% and 24.44%, respectively.

Our compound features also reduced the errors caused by ambiguous words, as shown in Figure

2(b), where the numbers of senses for a word are measured by the numbers of different POS tags it has in Penn Treebank. 12.1% of the errors on ambiguous words were reduced, comparing to 8.4% of the errors on unambiguous ones.

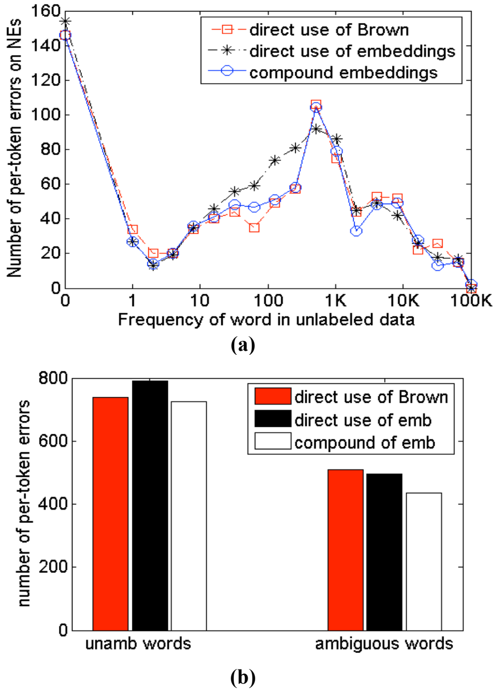

Figure 2: Errors incurred on words with different frequencies (a) and ambiguous words (b) in NER.

4.2 Linear separability of embeddings

Another reason for the good performances of compound features on NER is that they made linear models better separate ***named entities (NEs)*** and ***non-NEs***, which are more difficult to be linearly separated when embeddings are directly used as features. Here we designed an experiment to prove this. Based on training data of CoNLL2003, a classification task was built to tell whether a word belongs to NE or not. Linear SVM and a non-linear model ***Multilayer Perceptron (MLP)*** were used to build the classifiers. As shown in Table 7, when embeddings were directly used as features, MLP performed much better than linear SVM. And the linear model was under-fitting on this task since it had similar accuracies on both training set and development set. Above observations showed that linear models could not separate NEs and non-NEs well in the space of embeddings.

When clusters of embeddings were used as features, the accuracies of linear models increased even when there were only one or two non-zero features for each sample. At the same time the performances of MLP decreased because of the loss of information during clustering. The gaps between accuracies of linear models and non-linear ones decreased in the spaces of clusters, showing that cluster features are more suitable for linear models. At last, the compound features made the linear model out-perform all non-linear ones, since extra context information could be utilized.

Embeddings	Models	Accuracy
direct	linear	94.38
direct	MLP	96.87
cluster 1000	linear	95.31
cluster 1000	MLP	95.32
cluster 500+2500	linear	96.10
cluster 500+2500	MLP	96.02
compound	linear	**97.30**

Table 7: Performances of linear and non-linear models on development set with different embedding features.

5 Conclusion and perspectives

In this paper, we first introduced the idea of clustering embeddings and then proposed the compound features based on clustering, in order to overcome the disadvantages of the direct usage of continuous embeddings. Experiments showed that the compound features built on the same original word representation features (either embeddings or Brown clusters) achieve better performances on the same tasks. Further analyses showed that the compound features reduced errors on rare-words and ambiguous words and could be better utilized by linear models.

The usage of word embeddings also has some limitations, e.g. they are weak in capturing structural information of languages, which is necessary in NLP. In the future, we will research on task-specific representations for sub-structures, such as phrases and sub-trees based on word embeddings and documents representations (Xu et al., 2012).

Acknowledgments

We would like to thank Dr. Hua Wu, Haifeng Wang, Jie Zhou and Rui Zhang for many discussions and thank the anonymous reviewers for their valuable suggestions. This work was supported by National Natural Science Foundation of China (61173073), and the Key Project of the National High Technology Research and Development Program of China (2011AA01A207).

References

Bengio, Y., Ducharme, R., Vincent, P., and Jauvin, C. (2003). A neural probabilistic language models. The *Journal of Machine Learning Research*, 3:1137–1155.

Collobert, R. and Weston, J. (2008). A unified architecture for natural language processing: Deep neural networks with multitask learning. In *Proceedings of the 25th international conference on Machine learning*, pages 160–167. ACM.

Finkel, J., Grenager, T., and Manning, C. (2005). Incorporating non-local information into information extraction systems by gibbs sampling. In *Proceedings of the 43rd Annual Meeting on Association for Computational Linguistics*, pages 363–370. Association for Computational Linguistics.

Huang, F. and Yates, A. (2009). Distributional representations for handling sparsity in supervised sequence labeling. In *Proceedings of the Joint Conference of the 47th Annual Meeting of the ACL and the 4th International Joint Conference on Natural Language Processing of the AFNLP*: Volume 1-Volume 1, pages 495–503. Association for Computational Linguistics.

Koo, T., Carreras, X., and Collins, M. (2008). Simple semi-supervised dependency parsing. In *Proceedings of Association for Computational Linguistics*, pages 595–603. Association for Computational Linguistics.

Mnih, A. and Hinton, G. E. (2009). A scalable hierarchical distributed language model. *Advances in neural information processing systems*, 21:1081–1088.

Nivre, J., Hall, J., Kübler, S., McDonald, R., Nilsson, J., Riedel, S., and Yuret, D. (2007). The CoNLL 2007 shared task on dependency parsing. In *Proceedings of the CoNLL Shared Task Session of EMNLP-CoNLL*, pages 915–932.

Okazaki, N. (2007). Crfsuite: a fast implementation of conditional random fields (crfs). URL http://www.chokkan.org/software/crfsuite.

Schwenk, H. (2007). Continuous space language models. *Computer Speech & Language*, 21(3):492–518.

Sculley, D. (2010). Web-scale k-means clustering. In *Proceedings of the 19th international conference on World Wide Web*, pages 1177–1178. ACM.

Socher, R., Huang, E., Pennington, J., Ng, A., and Manning, C. (2011a). Dynamic pooling and unfolding recursive auto-encoders for paraphrase detection. *Advances in Neural Information Processing Systems*, 24:801–809.

Socher, R., Pennington, J., Huang, E., Ng, A., and Manning, C. (2011b). Semi-supervised recursive auto-encoders for predicting sentiment distributions. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing*, pages 151–161. Association for Computational Linguistics.

Täckström, O., McDonald, R., and Uszkoreit, J. (2012). Cross-lingual word clusters for direct transfer of linguistic structure. In *Proceedings of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 477–487, Montréal, Canada, June 3-8, 2012.

Turian, J., Ratinov, L., and Bengio, Y. (2010). Word representations: a simple and general method for semi-supervised learning. In *Annual Meeting-Association For Computational Linguistics*. Urbana, 51:61801.

Xu, Z., Chen, M., Weinberger, K., and Sha, F. An alternative text representation to TF-IDF and Bag-of-Words. In *Proceedings of 21st ACM Conf. of Information and Knowledge Management (CIKM)*, Hawaii, 2012.

Xue, N., Xia, F., Chiou, F., and Palmer, M. (2005). The penn chinese treebank: Phrase structure annotation of a large corpus. Natural Language Engineering, 11(2):207.