Multi-prototype Chinese Character Embedding

Yanan Lu[1], **Yue Zhang**[2], **Donghong Ji**[1]
Computer School, Wuhan University, China[1]
Singapore University of Technology and Design[2]
{luyanan, dhji}@whu.edu.cn, yuezhang@stud.edu.sg

Abstract

Chinese sentences are written as sequences of characters, which are elementary units of syntax and semantics. Characters are highly polysemous in forming words. We present a position-sensitive skip-gram model to learn multi-prototype Chinese character embeddings, and explore the usefulness of such character embeddings to Chinese NLP tasks. Evaluation on character similarity shows that multi-prototype embeddings are significantly better than a single-prototype baseline. In addition, used as features in the Chinese NER task, the embeddings result in a 1.74% F-score improvement over a state-of-the-art baseline.

Keywords: embedding, multi-prototype, Chinese character

1. Introduction

Learned from large-scale unlabeled data, embeddings offer distributed vector representations of words and phrases (Bengio et al., 2003; Mikolov et al., 2013). They have been applied as inputs to neural networks (Collobert and Weston, 2008; Socher et al., 2011; Kalchbrenner et al., 2014), and also used as additional features to improve linear models (Turian et al., 2010; Bansal et al., 2014; Guo et al., 2014b) for NLP tasks. The advantages of embeddings are two-fold. First, they are useful for reducing sparseness compared with discrete words and n-grams. Second, they contain automatically induced features.

Written as continuous sequences of characters, Chinese sentences do not have explicit word delimitation. As a result, similar to other Chinese NLP tasks, it is possible to create word embeddings based on automatically segmented Chinese sentences (Zhang et al., 2014b). Although segmentation errors can affect the quality, it has been shown that such word embeddings can improve Chinese NLP tasks such as parsing (Wu et al., 2013).

On the other hand, character information can also be used for NLP (Zhang et al., 2014a). Zhang et al. (2013) show that parsing can be improved by taking characters as inputs, jointly performing segmentation and syntax parsing. They show that character features can significantly improve the accuracies. Character features are also central to other Chinese NLP tasks such as character-based segmentation (Xue, 2003) and named entity recognition (NER (Chen et al., 2006)).

It is an interesting research question whether character embeddings can be useful for Chinese NLP. However, relatively little work has been reported on embedding Chinese characters, and limited effect has been observed (Sun et al., 2014). One possible reason is that the number of characters is much smaller compared to the number of words (10^4 vs 10^6), and the effect of character embedding on reducing sparsity can be limited. Also due to this reason, it has been shown that unsupervised clustering of characters does not improve in-domain segmentation (Liang, 2005).

Another possible reason, however, is that previous work did not take into account the polysemy of characters. One uniqueness of charactes, as compared with words, is that their use is highly flexible, and each character can have multiple meanings in different words. For example, the character "黄" can take at least 4 senses, including *color* in "黄色 (yellow)", *surname* in "黄氏 (Huang)", *failure* in "这事儿黄了 (The thing failed)" and *pornography* in "扫黄 (anti-pornography)". As a result, it is difficult to fully capture the character meaning by using a fixed embedding for each character. We address this issue by proposing a method for *multi-prototype* character embedding, which predicts the character sense together with its embedding given an input sentence.

Experiments show that multi-prototype embeddings can give better results than a single-prototype baseline on both character similarity evaluations and Chinese NER. For the NER task, character embeddings improves the accuracy of a state-of-the-art CRF model significantly.

2. Multi-prototype Character Embedding

Most Chinese characters have multiple senses, but take only one sense in a given context. The task of multi-prototype character embedding is to find a continuous vector representation for each Chinese character *sense*. We develop a multi-prototype word model by adapting an existing model for word embedding in the literature, making significant changes for the character embedding task.

There has been a line of work in training multi-prototype word embeddings (Huang et al., 2012; Guo et al., 2014a; Frey and Dueck, 2007; Tian et al., 2014; Chen et al., 2014). We adapt the Multi-Sense Skip-gram (MSSG) model (Neelakantan et al., 2014), which is based on the Skip-gram model (Mikolov et al., 2013). The Skip-gram model works by predicting a word vector using its context. To accommodate for multiple word senses, the MSSG model works by predicting the sense of the current word given its context, and then updating the current word sense vector and context vector.

In particular, each word w is associated with two types of vector representations, one being the output embedding $v(w)$, and the other being the context $\overline{v}(w)$. Given a special context, the sense of a word is predicted as the center

top K	1	5	10	20	50
one	33.1	26.3	23.7	21.0	17.7
fix	27.4	22.5	20.6	18.6	15.9
cilin	33.3	28.0	25.3	22.8	19.9
one_pos	36.0	30.2	27.1	24.0	20.1
fix_pos	34.7	29.5	26.7	24.0	20.2
cilin_pos	**39.6**	**33.3**	**30.2**	**27.5**	**23.7**

Table 1: Character similarity accuracy results.

of cluster that is closet to its average context vector. The MSSG model uses a clustering algothrim to group word sense, which similar to the k-means algorithm, and the cluster center is the average of the vector representation of all contexts which belong to that cluster.

We make three significant changes to MSSG for the *character* embedding task. First, we predict the sense of the current character given the context directly by using a neural model (Eq.2), rather than by finding the cluster center which is closed to average context vector. This method is empirically more accurate for our task. Second, characters are highly order-sensitive in forming words, we add position into the context by combining a character c with its position in obtaining $\overline{v}_c$, so each character in each position (relative to the current character position) has a embedding. This results in a *position-sensitive* variation of the Skip-gram model. In addition, the number of senses per character is induced from a lexicon rather than automatically. We call the Position-sensitive MSSG (PMSG).

As shown in Figure 1, given an input sentence, each character c is assigned to its context vector $\overline{v}(c)$; which is then used to predict both the character senses and output vectors via a neural network. Denoting the current character as c_0, its previous characters as c_{-1} and c_{-2}, respectively, and its next characters as c_1 and c_2, respectively, the model first computes a context vector $v_{context}(c_0)$ for the current character using

$$v_{context}(c_0) = \frac{1}{2 * r} \sum_{t \in [-r,0) \cup (0,r]} \overline{v}(c_t) \qquad (1)$$

Here $r = 2$ is the window size. $v_{context}(c_0)$ is used to predict the sense-specific vector $v(c_0, i)$ by

$$s_0 = \underset{i=1 \in [1,k(c_0)]}{\arg\max} \frac{1}{e^{-v(c_0,i)^T v_{context}(c_0)}} \qquad (2)$$

In the equation, $k(c_0)$ represents the number of senses (i.e. prototypes) of c_0 and $v(c_0, i)$ represents the embedding of the ith sense of c_0, and s_0 represents the sense index of current character c_0. Both the output and the context vectors are trained by predicting the context vectors using the Skip-gram model. AdaGrad (Duchi et al., 2011) is used with an initial learning rate of 0.025.

3. Experiments

We use 46 million online news sentences[1], and additionally the 7 million news sentences from Chinese Gigawords as the training data. All the characters occuring less than

[1]Downloaded from http://pullword.com/

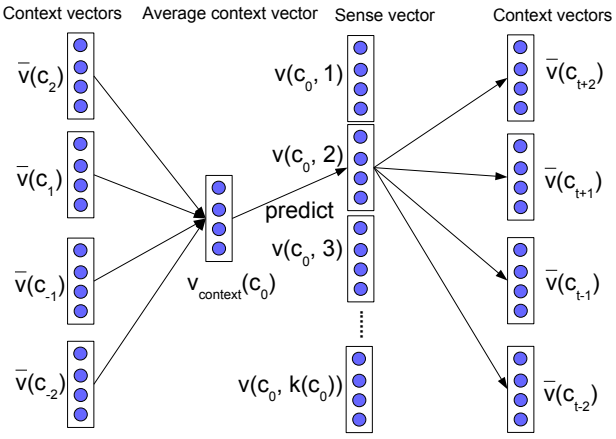

Figure 1: Multi-prototype Skip-gram model.

打 /v/Kb 这以后 (Since this)
虎 打 /v/Hi 武松 (Tiger beat Wusong)
那年，畹町 打 /v/Fa 出了对外开放的口号。
(Wan Ting put forward open-up announcement)

Figure 2: Example test data.

1,000 times are removed, resulting in 5,182 characters for embedding. We set $window = 5$ [2], and $dimensionality = 200$ when training the MSSG and PMSG models, and evaluate the embeddings on two tasks, by directly measuring character similarity, and application to Chinese NER.

3.1. Chinese Character similarity

We first evaluate the embeddings using a similarity measure, namely Precision@K (Qiu et al., 2011; Jin et al., 2012), which measures the precision of the K nearest neighbour by the cosine similarity. Because there are no publicly available datasets for evaluating polysemous characters, we construct a dataset based on the lexicon TongYiCiLin (*cilin*)[3], which contains 4,738 commonly used Chinese characters, and a semantically labeled Chinese corpus that consists of 10,000 sentences. There are 12 coarse-grand character categories, 97 classes, and 1200 fine-grand subclass in the *cilin* semantic hierarchy. We use the 97 classes as character senses, and the average sense number per character is 1.85.

We evaluate each character in its context in the labeled corpus in *cilin*, which covers 85 character classes. In addition, we add all the common surnames[4] to the surname class of *cilin*, and label all the surnames in the 10,000 sentences[5]. We randomly select a sentence for each sense of each character, obtaining 2,236 sentences, which contain 85 out of 97 classes and 1,553 out of 4,738 characters. The average sense number per character is 2.755. An example of the character "打" in our dataset is shown in Figure 2.

Table 1 shows the results when K is set to $1, 5, 10, 20, 50$, respectively. *one* represents the single-prototype baseline,

[2]here, $window$ is the parameter r of Eq.2

[3]Available at http://www.datatang.com/data/42306/

[4]Available at http://xing.911cha.com

[5]The original *cilin* surname class contains only 178 surnames, we extend it to 478.

system	Characters	top-5 similar characters
cilin_pos	黄: sense 1	红(red) 黑(black) 橙(orange) 色(colour) 蓝(blue)
	黄: sense 2	白(white;rivername) 赤(red) 滨(shore) 瓯(bowl) 湟(river name)
	黄: sense 3	白(white) 钯(palladium) 货(goods) 冲(hedge) 周(circumference)
	黄: sense 4	杨(surname) 刘(surname) 袁(surname) 周(surname) 邱(surname)
	叶: sense 1	茎(stem) 花(flower) 籽(seed) 菜(vegetables) 芽(bud)
	叶: sense 2	杨(surname) 黄(surname) 何(surname) 蒋(surname) 刘(surname)
one_pos	黄	姜(surname) 杨(surname) 白(white) 薛(surname) 褚(surname)
	叶	柳(willow) 姜(surname) 栗(surname) 杨(poplar) 菊(chrysanthemum)

Table 2: 5 nearest characters of "叶" and "黄" by the one_pos and cilin_pos systems.

Character	$C_i : i \in [-2, 2], C_iC_{i+1} : i \in [-2, 1]$
Sense	$S_i : i \in [-2, 2], S_iS_{i+1} : i \in [-2, 1]$
Embedding	$E_i : i \in [-2, 2], E_iE_{i+1} : i \in [-2, 1]$

Table 3: Features templates used in the CRF model.

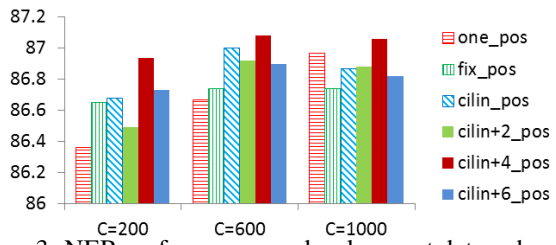

Figure 3: NER performance on development data, where C is the size of clusters.

fix represents the MSSG model trained with a fixed number of senses *s* for each character, which is set to 5[6]. And *cilin* represents the MSSG model trained with the sense number from *cilin* for each character. The *X_pos* rows show results of the PMSG models under the setting *X* ($X \in \{one, fix, cilin\}$). It can be seen from the table that the PMSG models perform significantly better than the MSSG models, showing the importance of position information on *character* embedding.

If the number of senses is fixed to 5 for each character, the performance can be similar to the single-prototype baseline. This is because different characters have very different numbers of senses. While the most frequent character (i.e. "干") has over 10 senses, a large number of low frequency characters have only one sense. By setting the number of senses according to the *cilin* corpus, the multi-prototype embeddings give the best results. However, there is not a universal standard on the number of senses for each character, due to variation in semantic granularity.

Table 2 gives the nearest characters for "黄" and "叶" by *one_pos* and *cilin_pos*. While *cilin_pos* can distinguish different senses of the characters "黄" and "叶", *one_pos* mixes different senses together. *cilin_pos* gives relatively more correct synonyms according to each sense. On the other hand, there is also noise. For example, for the character "黄", the second sense contains a mixture of river names and colors, while the third contains colors, metals and trading-related use of the character.

[6]We experiment with s=2-10 for the number of sense, and only report the best baseline (s=5) due to space limitations.

	F-Score
Chen	86.20
one C=600	86.96
one_pos C=1000	87.33
cilin+4_pos C=600	**87.94**

Table 4: NER performance on test data. where C is the size of clusters.

3.2. Application on Chinese NER

We apply the embeddings as features for Chinese NER. The experiments are conducted on the MSRA NER data in Sighan bakeoff 2006 (Ng and Kwong, 2006), which contains 46,364 training sentences and 4,365 test sentences. Following Chen et al. (2006), we use the first 37,000 sentences of the MSRA training data for training and the last 9,364 sentences for development.

We take the character-based CRF model of Chen et al. (2006) as our baseline, which is the best CRF model in the closed test. Chen et al. (2006) show that character-based CRF is better than word-based CRF. We use their feature set as the base set, also apply sense and embedding cluster features as shown in Table 3. C_i is the i_{th} character in the sentence, S_i is the i_{th} character sense, E_i is the i_{th} embedding. The subscript 0 denotes the current position, -1 denotes the previous position and 1 denotes the next position. We use as embedding features character embedding clusters by the K-means algorithm.

As a first development test, we study how the number of senses and size of clusters affect NER performance. The dimensionality of embeddings is set to 200, and all the embeddings are trained using the PMSG model. The *one_pos* system utilizes the baseline and embedding cluster feature templates in Table 3. The other systems use the baseline, sense and character embedding cluster feature templates. *cilin*, *one_pos*, *fix_pos* and *cilin_pos* have the same meanings as Table 1. To test the influence of sense counts, *cilin+N* represents the number of senses per character *cilin* plus *N* if it is greater than 1.

The results are shown in Figure 3, from which we obtain the best parameters for each system. In the figure, the sense count has a significant influence on the development accuracies, and multi-prototype embeddings are generally better than single-prototype embeddings.

Table 4 shows the test results. The character embeddings improve the F-score over the baseline by a large improvement in recall, and a slight drop in precision. The multi-prototype embeddings outperform the single-prototype em-

beddings by 0.98% F-score (significance at $p < 0.01$). Error analysis shows that the improvement are achieved by significantly better recall. This directly benefits from character similarity using embeddings. For example, the recall on person names are largely higher when multi-prototype embeddings are used.

4. Conclusion

We studied embeddings of Chinese characters, comparing a novel multi-prototype embedding method with a single-prototype embeddings baseline on direct similarity measures and an application to Chinese NER. Experiments show that multi-prototype embeddings can induce significantly better character synsets compared to single-prototype, while position information of context characters and the number of senses for each character are crucial to the training of embeddings. Positive results are obtained on both similarity and NER, showing that character embedding can be useful to Chinese NLP.

5. Acknowledgements

We thank the anonymous reviewers for their constructive comments, which helped to improve the paper. This work is supported the State Key Program of National Natural Science Foundation of China (Grant No.61133012), the National Natural Science Foundation of China (Grant No.61373108), the Singapore Ministry of Education (MOE) AcRF Tier 2 grant T2MOE201301, SRG ISTD 2012 038 from Singapore University of Technology and Design.

6. Bibliographical References

Bansal, M., Gimpel, K., and Livescu, K. (2014). Tailoring continuous word representations for dependency parsing. In *Proceedings of the Annual Meeting of the Association for Computational Linguistics*.

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

Chen, A., Peng, F., Shan, R., and Sun, G. (2006). Chinese named entity recognition with conditional probabilistic models. In *Proc. Fifth SIGHAN Workshop Chinese Language Processing*, pages 173–176.

Chen, X., Liu, Z., and Sun, M. (2014). A unified model for word sense representation and disambiguation. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 1025–1035.

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.

Duchi, J., Hazan, E., and Singer, Y. (2011). Adaptive subgradient methods for online learning and stochastic optimization. *The Journal of Machine Learning Research*, 12:2121–2159.

Frey, B. J. and Dueck, D. (2007). Clustering by passing messages between data points. *science*, 315(5814):972–976.

Guo, J., Che, W., Wang, H., and Liu, T. (2014a). Learning sense-specific word embeddings by exploiting bilingual resources. In *Proceedings of COLING*, pages 497–507.

Guo, J., Che, W., Wang, H., and Liu, T. (2014b). Revisiting embedding features for simple semi-supervised learning. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 110–120.

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

Jin, P., Carroll, J., Wu, Y., and McCarthy, D. (2012). Distributional similarity for chinese: Exploiting characters and radicals. *Mathematical Problems in Engineering*, 2012.

Kalchbrenner, N., Grefenstette, E., and Blunsom, P. (2014). A convolutional neural network for modelling sentences. *Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics*, June.

Liang, P. (2005). Semi-supervised learning for natural language. Master's thesis, Massachusetts Institute of Technology.

Mikolov, T., Sutskever, I., Chen, K., Corrado, G. S., and Dean, J. (2013). Distributed representations of words and phrases and their compositionality. In *Advances in Neural Information Processing Systems*, pages 3111–3119.

Neelakantan, A., Shankar, J., Passos, A., and McCallum, A. (2014). Efficient non-parametric estimation of multiple embeddings per word in vector space. In *Proceedings of EMNLP*.

Hwee Tou Ng et al., editors. (2006). *Proceedings of the Fifth SIGHAN Workshop on Chinese Language Processing*. Association for Computational Linguistics, Sydney, Australia, July.

Qiu, L., Wu, Y., and Shao, Y. (2011). Combining contextual and structural information for supersense tagging of chinese unknown words. In *Computational Linguistics and Intelligent Text Processing*, pages 15–28. Springer.

Socher, R., Lin, C. C., Manning, C., and Ng, A. Y. (2011). Parsing natural scenes and natural language with recursive neural networks. In *Proceedings of the 28th International Conference on Machine Learning (ICML-11)*, pages 129–136.

Sun, Y., Lin, L., Tang, D., Yang, N., Ji, Z., and Wang, X. (2014). Radical-enhanced chinese character embedding. *arXiv preprint arXiv:1404.4714*.

Tian, F., Dai, H., Bian, J., Gao, B., Zhang, R., Chen, E., and Liu, T.-Y. (2014). A probabilistic model for learning multi-prototype word embeddings. In *Proceedings of COLING*, pages 151–160.

Turian, J., Ratinov, L., and Bengio, Y. (2010). Word representations: a simple and general method for semi-supervised learning. In *Proceedings of the 48th Annual Meeting of the Association for Computational Linguis-*

tics, pages 384–394. Association for Computational Linguistics.

Wu, X., Zhou, J., Sun, Y., Liu, Z., Yu, D., Wu, H., and Wang, H. (2013). Generalization of words for chinese dependency parsing. *IWPT-2013*.

Xue, N. (2003). Chinese word segmentation as character tagging. *Computational Linguistics and Chinese Language Processing*, 8(1):29–48.

Zhang, M., Zhang, Y., Che, W., and Liu, T. (2013). Chinese parsing exploiting characters. In *ACL (1)*, pages 125–134.

Zhang, M., Zhang, Y., Che, W., and Liu, T. (2014a). Character-level chinese dependency parsing. In *Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1326–1336, Baltimore, Maryland, June. Association for Computational Linguistics.

Zhang, Q., Kang, J., Qian, J., and Huang, X. (2014b). Continuous word embeddings for detecting local text reuses at the semantic level.