Learning Character Representations for Chinese Word Segmentation

Xiaodong Liu, Kevin Duh and Yuji Matsumoto
Nara Institute of Science and Technology
8916-5 Takayama, Ikoma, Nara, Japan
{xiaodong-l,kevinduh,matsu}@is.naist.jp

Tomoya Iwakura
Fujitsu Laboratories Ltd.
1-1, Kamikodanaka 4-chome, Nakahara-ku, Kawasaki, Japan
iwakura.tomoya@jp.fujitsu.com

Abstract

We propose a simple yet effective semi-supervised method for improving Chinese Word Segmentation. Our method is based on learning generalizable vector and cluster representations of variable-length character sequences from large unlabeled data, which is then incorporated into a sequence labeling model with the passive-aggressive algorithm as features. We achieve state-of-the-art results on the SIGHAN2005 Bakeoff and show significant improvements on Out-Of-Vocabulary (OOV) words, a long-standing challenge in word segmentation.

1 Motivation

The problem of OOV words plagues all Natural Language Processing (NLP) systems. Words that are not observed in the training data often deteriorate accuracy, limiting the usability of NLP in open domains. Recently, methods for learning word representations from large unlabeled text have been shown to be effective for this problem. The idea is to learn generalizable features or clusters of words in order to tie OOV words with the words observed in the training data. Positive results are shown in Named Entity Recognition, POS tagging [10], and dependency parsing [3].

However, learning *word* representations is challenging for writing systems that do not have clear word delimiters (e.g. Chinese, Japanese). A reasonable solution is to resort to learning *character* representations, though it is not obvious whether single characters contain sufficient information.

In this work, we perform a *comprehensive evaluation* of character representations in the context of Chinese Word Segmentation, and propose a technique that *learns representations for variable-length character n-grams* using Neural Language Models [5] and Brown Clustering [1]. This representation learning approach is in contrast to previous works, which focus on mining *point-wise mutual information* or *accessor variety* features from unlabeled text [8, 7, 12]. Our method achieves state-of-the-art results on the SIGHAN2005 Bakeoff and improves recall for OOV words, the main factor impacting segmentation accuracies [2]. We also demonstrate that our method improves accuracy in the open domain scenario, using the SIGHAN2010 Bakeoff data.

2 Learning Representations

2.1 Proposed Framework

Our method is summarized by Figure 1. By learning vector and cluster representations of variable-length character sequences from large unlabeled text, we seek for features that generalize across both observed and OOV words. First, the raw text (without word delimiters) is split into n-gram character chunks. For example, given a sentence $s = [c_0c_1c_2c_3c_4]$ with characters c_i, we split it into 1-gram character chunks $[c_0, c_1, c_2, c_3, c_4]$, 2-gram character chunks $[c_0c_1, c_1c_2, c_2c_3, c_3c_4]$, and 3-gram character chunks $[c_0c_1c_2, c_1c_2c_3, c_2c_3c_4]$.

Next, we run standard "word"-based representation learning algorithms (e.g. Neural Language Models and Brown Clustering), treating each of these character n-grams as a "word". Finally, features derived from such representations are merged and plugged into a word segmentation algorithm.

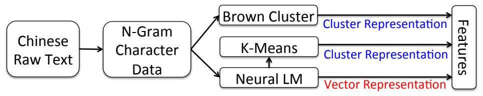

Figure 1: Proposed multi-scale and ensemble framework for learning representations.

Our method can be considered as a *multi-scale* and *ensemble* approach to learning character representations. By allowing *multi-scale* character n-gram sequences, we increase the chance of learning representations for meaningful sub-units of words. Of course, some of the character n-grams produced in this way may be nonsense "sub-units", but by using an *ensemble* of multiple learning representation algorithms, we decrease the risk of settling on an unreliable feature. This approach naturally addresses the issue of learning representations in the case where raw text lacks word delimiters. Two types of character representations employed here are described in next subsection.

2.2 Vector-based Character Representations

The vector-based approach to learning representations focuses on embedding a word or character n-gram as a low-dimensional real-valued vector. The goal is to place similar character n-grams into nearby points in the vector space. Let $v \in \mathcal{R}^d$ be a d-dimensional vector representation of the character n-gram we would like to learn, and v_c be the vector representation of a character n-gram in the sentence context. Following Neural LM work by [5], we optimize v by maximizing the following objective:

$$\sum_c \log p(v_c|v) = \sum_c \log \left[\frac{\exp(v_c^T \cdot v)}{\sum_w \exp(v_w^T \cdot v)} \right] \tag{1}$$

Basically, the equation says that we want to learn a probabilistic model $p(v_c|v)$ that gives high probability to all v_c occurring in a window[1] around v, and that this probabilistic model is defined by a dot-product $v_c^T \cdot v$ put through as softmax activation function. The normalizing denominator in the softmax ($\sum_w \exp(v_w^T \cdot v)$) sums over all character n-grams (i.e. vocabulary size), in effect penalizing words that are not in v's context. The objective in Eq. 1 is optimized jointly for all v in the large unlabeled text to generate vector representations for all character n-grams. Following [10], these vector features are normalized and scaled in order to maintain a balanced contribution compared with other binary features.

2.3 Cluster-based Character Representations

In a cluster-based representation, we first cluster all character n-grams using some objective and give the same feature to those that share the same cluster. In contrast to real-valued vectors, these are sparse and discrete features. While discrete features may be easier to integrate into existing NLP systems, we loss the real vector's ability to discern graded similarities. Our first cluster-based representation is K-Means clustering applied on the vectors learned by Equation 1. This kind of two-stage approach that clusters vectors learned from Neural LMs has been shown successful [11].

[1]We use the previous two and next two character n-grams in our experiments. If using only the next character n-gram, the model becomes a bigram language model.

Our second cluster-based representation is the widely-used Brown clustering algorithm [1] trained on large unlabeled raw text. Brown clustering is a hierarchical clustering algorithm which groups words (character n-grams in our case) to maximize the mutual information between clusters in a class-based bigram language model.

3 Experimental Results

3.1 Data and Setup

We evaluate our method on two simplified Chinese corpora, the Peking University (PKU) corpus, which includes 1.1M training words and 104K test words with OOV Rate 0.058, and the Microsoft Research (MSR) corpus, which includes 2.37M training words and 107K test words with OOV Rate 0.026, provided in the SIGHAN2005 Bakeoff[2]. Standard train/test splits are used in training and testing our baseline character-based CRF segmenter, which is implemented in CRFSuite [6] using passive-aggressive algorithm. The CRF segments by tagging each character with one of four tags $\{B, I, E, S\}$: B, I, E denote the beginning, internal, and end character of a multi-character word, respectively, and S indicates a single character word. Given a current character c_i with a context ...$c_{i-1}c_ic_{i+1}$..., the baseline feature templates are [9]:

- Char Unigram: c_s (i-3$<$ s $<$ i+3)
- Char Bigram: c_sc_{s+1} (i-3$<$ s $<$ i+2)
- Identical: $c_s = c_{s+1}$ (i-2$<$ s $<$ i+2)
- Identical skip: $c_s = c_{s+1}$ (i-4$<s<i$+2)

Our character representations are appended as extra features.For target character c_s, we extract Neural LM, K-Means, and Brown features based on the representations trained on 1-gram, 2-gram, and 3-gram character chunks. The Neural LMs are vectors in $R^d, d = 100$, trained using `word2vec`[3]. K-Means[4] with $k = 100$ clusters is trained on these Neural LM vectors. For Brown clustering[5], we employ the full bit string and 4-6 prefix as features. To train our character representations, we use the simplified Chinese data in Chinese Gigaword v2[6]. This data contains a total of 559M characters; our character n-gram data has 6K unigram types, 1.4M bigram types, and 7.3M trigram types. We also obtain Pointwise Mutual Information (PMI) features from Gigaword for comparison; this is an effective semi-supervised technique [8].

3.2 Main Results

Table 1 summarizes our segmentation results. On the PKU test set, baseline F-value of 94.6% is improved to 95.5% when using the proposed method; on the MSR test set, baseline F-value of 96.6% is increased to 97.0%. Both of these results outperform the best systems at Bakeoff2005 and are also competitive with the current state-of-the-art.[7] Notably, our features are more effective than PMI, a common approach to exploit unlabeled data.

More interesting are our OOV recall results, which improved significantly from 73.2% to 80.9% on PKU, and from 73.0% to 75.3% on MSR. In Chinese, compounding is one of the main mechanisms whereby new words are generated. We find that many of the OOV words corrected by our proposed method are such compound words, such as: 专利(patent)法(law) and 沙丁(sardine ”transliteration”)鱼(fish).

To see the effect of our *multi-scale* and *ensemble* approach, note that (1) for all representations, while 1-gram, 2-gram, and 3-gram each helped individually, using all of them tend to give the best results; (2) there appears to be no winner among Neural LM, Brown Cluster, and Neural LM/K-Means individually; each of them tend to give the same amount of improvement, yet when used together, their improvements are additive, implying that they provide diverse information.

[2]http://www.sighan.org/bakeoff2005/

[3]https://code.google.com/p/word2vec

[4]http://users.eecs.northwestern.edu/∼wkliao/K-Means/

[5]https://github.com/percyliang/brown-cluster

[6]http://catalog.ldc.upenn.edu/LDC2005T14

[7]To the best of our knowledge, the current best F-value for PKU is 95.55% by [7], which mines frequent substrings from both Gigaword and *test* data. For MSR, the current best (97.4%) appears to be [9]; their contribution is a joint word detection and segmentation model and it could conceivably be combined with our features.

System	PKU				MSR			
	Recall	Prec	F-val	OOV	Recall	Prec	F-val	OOV
baseline	94.1	95.1	94.6	73.2	96.4	96.7	96.6	73.0
+PMI	94.4	95.6	95.0	78.4	96.5	96.7	96.6	71.8
+1-gram Neural LMs	94.1	95.4	94.7	78.0	96.4	96.8	96.6	73.3
+2-gram Neural LMs	94.2	95.5	94.8	78.6	96.5	96.8	96.7	73.8
+3-gram Neural LMs	94.3	95.5	94.9	79.0	96.5	96.9	96.8	74.0
+All Neural LMs (ANLM)	94.3	95.6	94.9	79.5	96.5	96.8	96.7	73.6
+1-gram K-Means	94.1	95.3	94.7	78.1	96.4	96.8	96.6	73.7
+2-gram K-Means	94.2	95.5	94.9	79.1	96.6	96.9	96.7	74.8
+3gram K-Means	94.5	95.6	95.0	80.1	96.6	96.9	96.8	74.3
+All K-Means (AKM)	94.5	95.7	95.1	80.4	96.8	96.9	96.8	**75.8**
+1-gram Brown Cluster	94.0	95.3	94.7	78.2	96.5	96.8	96.6	73.7
+2-gram Brown Cluster	94.6	95.6	95.1	79.5	96.7	96.9	96.8	72.6
+3-gram Brown Cluster	94.6	95.7	95.1	80.5	96.6	96.9	96.8	74.4
+All Brown Cluster (ABC)	94.7	95.7	95.2	80.5	96.8	96.9	96.8	74.4
+ANLM+ ABC +PMI	94.8	95.9	95.4	80.8	96.9	96.9	96.9	74.3
+AKM + ABC + PMI	94.9	**96.1**	**95.5**	**80.9**	**97.0**	**96.9**	**97.0**	75.3
Chen05	**95.3**	94.6	95.0	63.6	96.9	95.2	96.0	37.9
Tseng05	94.6	95.4	95.0	78.7	96.2	96.6	96.4	71.7

Table 1: Results on PKU and MSR corpora (overall precision, recall, F-value, & OOV recall). *All Neural LMs* (ANLM) and *All Brown cluster* (ABC) indicates all of the 1-gram, 2-gram and 3-gram features. Tseng05 and Chen05 are the best systems in PKU and MSR at Bakeoff2005, respectively.

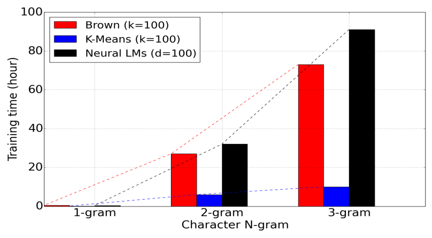

Figure 2: Training time on Intel 2.9GHz Xeon.

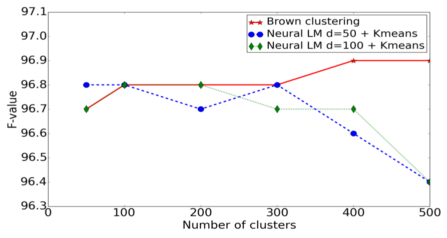

Figure 3: F-value vs. #cluster (MSR corpus).

The training time for learning representations is shown in Figure 2. While it is practically feasible to train 3-grams on commodity hardware and achieve good results, the complexity does grow with larger n-grams due to the larger number of ”word” types.

We have found that our method is not very sensitive to hyper-parameters, such as number of clusters. Figure 3 shows that for cluster sizes k=50 to 300, the F-value is relatively constant. It is only at larger k (e.g. 500) where we see slight differences in performance.

3.3 Open Domain Results

We are interested in how our system performs in an open domain setting. So here we took the PKU model trained on SIGHAN2005 and applied it to the multi-domain test set of SIGHAN2010 [14]. Notably, we do not re-train the character representations nor the CRF in order to simulate a true open domain setting. From Table 2, we see that the our proposed approach drastically improves both F-value (by 1% or more) and OOV recall (by 3% or more) in all test domains.

4 Conclusions

We have explored how a multi-scale and ensemble approach to learning character representations can improve Chinese Word Segmentation. By combining Neural LM, Brown, and K-Means features trained on various n-gram character chunks, we achieved state-of-the-art F-value and OOV recall on

	Domains (F-Value/OOV Recall)			
	Literature	Computer	Medicine	Finance
Baseline	91.7/64.1	90.5/67.7	90.9/67.4	94.4/80.5
Proposed	93.1/67.9	92.7/74.7	92.7/73.4	95.3/83.5

Table 2: Open Domain test results on Bakeoff2010. Proposed system uses all n-gram features of Brown clustering and K-Means.

Bakeoff2005, and demonstrated large gains in open domain scenarios. As future work, we plan to compare and combine with semi-supervised systems that utilize unlabeled data differently, e.g. co-training and co-regularization [13, 15]. Another technique worth trying is to train variable-length representations from automatically-segmented corpus, then integrate them as features to a semi-Markov model. Finally, it is interesting to consider hierarchical approaches to learning sub-word representations [4].

References

[1] P. F. Brown, P. V. deSouza, R. L. Mercer, V. J. D. Pietra, and J. C. Lai. Class-based n-gram models of natural language. *Comput. Linguist.*, 18(4):467–479, Dec. 1992. ISSN 0891-2017. URL `http://dl.acm.org/citation.cfm?id=176313.176316`.

[2] C. Huang and H. Zhao. Chinese word segmentation: A decade review. *Journal of Chinese Information Processing*, 21(3):8–20, 2007.

[3] T. Koo, X. Carreras, and M. Collins. Simple semi-supervised dependency parsing. In *Proceedings of ACL-08: HLT*, pages 595–603, Columbus, Ohio, June 2008. Association for Computational Linguistics.

[4] T. Luong, R. Socher, and C. Manning. Better word representations with recursive neural networks for morphology. In *Proceedings of the Seventeenth Conference on Computational Natural Language Learning*, pages 104–113, Sofia, Bulgaria, August 2013. Association for Computational Linguistics. URL `http://www.aclweb.org/anthology/W13-3512`.

[5] T. Mikolov, I. Sutskever, K. Chen, G. Corrado, and J. Dean. Distributed representations of words and phrases and their compositionality. In *NIPS*, 2013.

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

[7] M. Shen, D. Kawahara, and S. Kurohashi. Chinese word segmentation by mining maximized substrings. In *Proceedings of the Sixth International Joint Conference on Natural Language Processing*, pages 171–179, Nagoya, Japan, October 2013. Asian Federation of Natural Language Processing. URL `http://www.aclweb.org/anthology/I13-1020`.

[8] W. Sun and J. Xu. Enhancing chinese word segmentation using unlabeled data. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing*, pages 970–979. Association for Computational Linguistics, 2011.

[9] X. Sun, H. Wang, and W. Li. Fast online training with frequency-adaptive learning rates for chinese word segmentation and new word detection. In *Proceedings of the 50th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 253–262, Jeju Island, Korea, July 2012. Association for Computational Linguistics. URL `http://www.aclweb.org/anthology/P12-1027`.

[10] J. Turian, L. Ratinov, and Y. Bengio. Word representations: a simple and general method for semi-supervised learning. In *Proceedings of the 48th Annual Meeting of the Association for Computational Linguistics*, pages 384–394. Association for Computational Linguistics, 2010.

[11] X. Wu, J. Zhou, Y. Sun, Z. Liu, D. Yu, H. Wu, and H. Wang. Generalization of words for chinese dependency parsing. *IWPT-2013*, page 73, 2013.

[12] X. Zeng, D. F. Wong, L. S. Chao, and I. Trancoso. Co-regularizing character-based and word-based models for semi-supervised chinese word segmentation. In *51st Annual Meeting of the Association for Computational Linguistics*, 2013.

[13] L. Zhang, H. Wang, X. Sun, and M. Mansur. Exploring representations from unlabeled data with co-training for Chinese word segmentation. In *Proceedings of the 2013 Conference on Empirical Methods in Natural Language Processing*, pages 311–321, Seattle, Washington, USA, October 2013. Association for Computational Linguistics. URL `http://www.aclweb.org/anthology/D13-1031`.

[14] H. Zhao and Q. Liu. The cips-sighan clp 2010 chinese word segmentation bakeoff. In *Proceedings of the CIPS-SIGHAN Joint Conference on Chinese Language Processing*, 2010.

[15] X. Zheng, H. Chen, and T. Xu. Deep learning for Chinese word segmentation and POS tagging. In *Proceedings of the 2013 Conference on Empirical Methods in Natural Language Processing*, pages 647–657, Seattle, Washington, USA, October 2013. Association for Computational Linguistics. URL `http://www.aclweb.org/anthology/D13-1061`.