Addressing Domain Adaptation for Chinese Word Segmentation with Instances-based Transfer Learning

Yanna Zhang, JinAn Xu*, Guoyi Miao, Yufeng Chen and Yujie Zhang

School of Computer and Information Technology, Beijing Jiaotong University
{yannazhang,jaxu,gymiao,chenyf,yjzhang}@bjtu.edu.cn

Abstract. Recent studies have shown effectiveness in using neural networks for Chinese Word Segmentation (CWS). However, these models, constrained by the domain and size of the training corpus, do not work well in domain adaptation. In this paper, we propose a novel instance-transferring method, which use valuable target domain annotated instances to improve CWS on different domains. Specifically, we introduce semantic similarity computation based on character-based n-gram embedding to select instances. Furthermore, training sentences similar to instances are used to help annotate instances. Experimental results show that our method can effectively boost cross-domain segmentation performance. We achieve state-of-the-art results on Internet literatures datasets, and competitive results to the best reported on micro-blog datasets.

Keywords: Chinese Word Segmentation, Domain Adaptation, Instance-Transferring, Neural Network.

1 Introduction

Chinese word segmentation (CWS) is a preliminary and important task for many Chinese natural language processing (NLP) tasks. Recently, neural word segmentation has shown promising progress [1, 2, 3, 4]. However, these neural network models, mainly trained by supervised learning, rely on massive labeled data. In recent years, large-scale human annotated corpora mainly focus on domains like newswire, the word segmentation performance trained on these corpora usually degrades significantly when the test data shift from newswire to micro-blog texts and Internet literatures [5, 6]. Such a problem is well known as domain adaptation [7]. Usually, the domain of training and testing data is called source and target domain respectively.

There are severe challenges to solving the problem of domain adaptability. On one hand, the In-Vocabulary (IV) word in different domains has different contexts and semantics, which affect the performance of word segmentation on target domain. On the other hand, many domain-related words in target domain that rarely appear in source domains. Therefore, Out-of-Vocabulary (OOV) word recognition becomes an important problem. Take the sentence "普泓 上人 点头， 又 看 了 鬼厉 一 眼， 转 身 便 要 走 了 出去 。" for example, the word "鬼厉" is the name of a person that often appears in literatures-related domains while seldom appears in other domains.

Table 1. The People's Daily (PD) is source data, the Micro-blog and "Zhuxian" (ZX) are target data. "Words" is the amount of words in different corpus. "Cover" indicates the percentage of words in target domain that can be covered by source domain.

Datasets	Words	Cover
PD	21653284	-
ZX	96934	80.48%
Micro-blog	421166	70.62%

As listed in Table 1, we count up the numbers of words in different domains and find that about 30% of the words in Micro-blog texts (target domain) are not available in People's Daily (source domain). As a result, it is necessary to develop new methods for addressing the problem of domain adaptability.

Instance-based transfer learning proves to be an excellent fit for alleviating the above two problems [8, 21]. In this paper, we propose a new instance-based transfer learning method, which effectively alleviate the OOV words recognition problem by adding the similar target domain labeled data to the training data. First, we obtain samples from large-scale unlabeled target domain data according to sampling modules, which introduces character-based n-gram embedding to calculate the similarity between two sentences. Second, we train an initial segmentation model with source domain data to annotate samples and then revise the segmentation result with the help of training data. Our proposed method is simple, efficient and effective, giving average 3.5 % the recall of OOV words on target domain data.

The contributions of this paper could be summarized as follows.

- The semantic similarity is first introduced to address the problem of adaptation domain for CWS, which is effective to select useful target domain instances (Section 3.2).
- We propose a new semantic similarity calculation method with the help of character-based n-gram embedding, which incorporates rich contextual information (Section 3.2).
- Training sentences similar to instances are used to construct partial annotated instances, which rectify partial improper annotation caused by word segmentation model (Section 3.3).

2 Related Work

Our work focus on domain adaptation for neural word segmentation, mainly using the character-based n-gram contextual information.

Neural Word Segmentation. Most modern CWS methods treated CWS as a sequence labeling problems. There has been a recent shift of research attention in the word segmentation literature from statistical methods to deep learning. Pei et al. [1] used Convolutional Neural Network (CNN) to capture local information within a fixed size window and proposed a tensor framework to capture the information of

previous tags. Chen et al. [2] proposed Gated Recursive Neural Network (GRNN) to model feature combinations of context characters. Subsequently, Cai et al. [3] proposed a gated combination neural network which can utilize complete segmentation history. Moreover, Chen et al. [4] adopted an adversarial multi-criteria learning method to integrate shared knowledge from multiple heterogeneous segmentation criteria. However, these neural word segmentation methods rely on a large-scale labeled data which is usually expensive and tends to be of a limited amount.

Transfer Learning in CWS. Transfer learning aims to learn knowledge from different source domains to enhance the word segmentation performance in a target domain. Transfer learning includes domain adaptation, which has been successfully applied to many fields. In particular, several methods have been proposed for solving domain adaptation problem in CWS. Lin et al. [21] proposed a simple yet effective instance-based transfer learning method, which employs a double-selection process to reduce the impact of harmful data in the source. Similar to Lin et al., this paper uses instance-based transfer learning method to solve the domain adaptation problem for segmentation. However, unlike his data selection method, we select samples with abundant target domain feature with the help of use the similarity calculation. Liu et al. [6] considered self-training and character clustering for domain adaptation. Liu et al. [8] proposed a variant CRF model to leverage both fully and partially annotated data transformed from different sources of free annotations consistently. Zhang et al. [9] used domain specific tag dictionaries and only unlabeled target domain data to improve target-domain accuracies. Furthermore, Huang et al. [10] proposed a novel BLSTM-based neural network model which incorporates a global recurrent structure designed for modeling boundary features dynamically. Xu et al. [11] trained a teacher model on source corpora and then use the learned knowledge to initialize a student model, which can be trained on target data by a weighted data similarity method.

The Contextual Information. During CWS, a character or a word usually has different meaning in different positions, which is called the ambiguity of the character or the word. Blitzer et al. [12] indicated that much of ambiguity in character or word meaning can be resolved by considering surrounding words. Inspired by the above, many scholars tried to solve some NLP problems by adding contextual information. Choi et al. [13] used the context-dependent word representation to improve the performance of machine translation. Qin et al. [14] presented a neural model with context-aware character-enhanced embeddings to address implicit discourse relation recognition task. Bao et al. [15] integrated the contextualized character embedding into neural word segmentation to capture the useful dimension in embedding for target domain. Zhou et al. [16] proposed word-context character embeddings (WCC), which contain the label distribution information. Motivated by that, we concatenate the contextual embeddings to calculate the similarity of two sentences.

3 Methods

3.1 Instances-based Transfer Learning for CWS

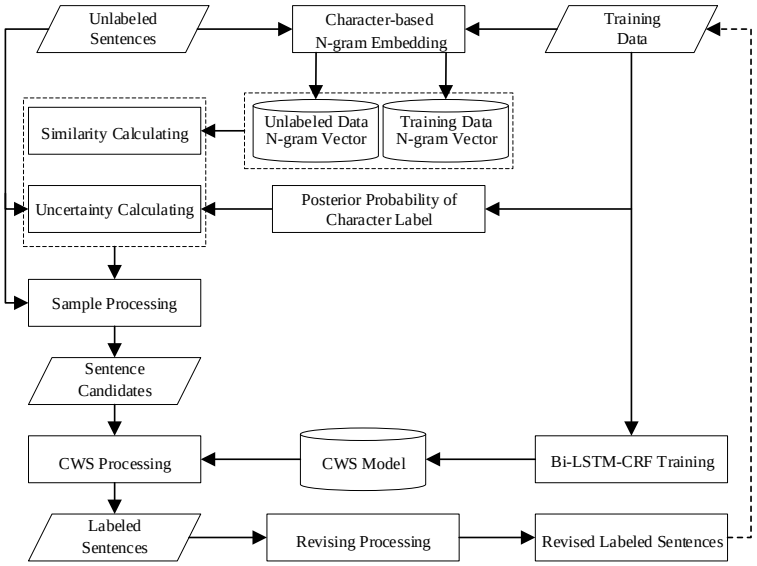

Fig. 1. Framework of instances-based transfer learning training process for CWS.

Instances-based transfer learning can be regarded as a learning method which continuously increases training data. Our method consists of two main stages. In the first stage, we obtain instances from large-scale unlabeled target domain sentences according to sampling strategy (See section 3.2). In the second stage, we train an initial segmentation model with source domain data to annotate instances and then revise the segmentation result (See section 3.3). These revised labeled instances are added to the training data to re-train the model. Fig. 1 illustrates the framework of our instances-based transfer learning training process for CWS. Finally, we can obtain the optimal word segmentation model by continuous iteration, which apply to target domain word segmentation task.

3.2 Obtaining Unlabeled Target Domain Instances

Character-based n-gram Embedding. Each character can be represented as a d-dimensional vector by using word2vec [18] toolkit. In order to incorporate the contextual information into vector representation of a character, we define the character-based n-gram embedding. As it is believed that the contextual information from a window size of 5 characters may be sufficient for CWS [19], we employ a max window size of 5 characters to generate the character-based n-gram embedding. For each character x_i in a given input sentence $\mathrm{X} = (x_1, x_2, \ldots, x_n)$, it's 5-gram embedding is

$e_i^5 = [x_{i-2}, x_{i-1}, x_i, x_{i+1}, x_{i+2}]$, which refers to a concatenate of 5 character embeddings around i-th character. We set the window size to 1, 3 and 5 respectively, and get three kinds of representations of this character.

The Similarity between Two Sentences. To extract useful samples from the large-scale unlabeled target domain corpus, we propose to calculate the character-based semantic similarity between training data and unlabeled target domain data. Formally, the character-based semantic similarity is defined as follows:

$$\mathrm{Sim}(s, \tilde{s}) = \sum_{j=1}^{3} w_j cos < \vec{s}, \vec{\tilde{s}} > \quad (1)$$

where $\vec{s}$ denotes target domain sentence vector and $\vec{\tilde{s}}$ indicates source domain sentence vector. $cos < \vec{s}, \vec{\tilde{s}} >$ denotes the Cosine distance between two vectors. w_j is the weight of each kind of sentence vector representation. We define the n-gram sentence vector $\vec{s}$ as:

$$\vec{s} = \frac{1}{n} \sum_{i=1}^{n} e_i^T \quad (2)$$

where e_i^T represents the character-based n-gram embedding of the i-th character. Here are three kinds of sentence vectors due to the different values of T, which is window size. n denotes the length of a sentence.

The Uncertainty of Annotation. Higher the uncertainty of an instance's annotation, more useful features the instance contains. In our work, therefore, we choose the instances with higher uncertainty in annotation task.

Additionally, the word segmentation can be represented as a two-class problem according to one factor, that is, whether the character is the right boundary of a word in a sentence. Specifically, the labels B, M, E, S can be divided into two categories: B and M can be grouped together, denoted as N, indicating that the character isn't the right boundary of a word; Also, E and S can be grouped and denoted as Y, indicating that the character is the right border of a word and need to be divided.

We use information entropy to measure the uncertainty of every character label and it is formally denoted as:

$$\mathrm{E(c)} = -\sum_{i=N,Y} p_i log_2 p_i \quad (3)$$

where, c is a character in instances, $p_N = p_B + p_M$, $p_Y = p_E + p_S$, p_B represents the posterior probability that c is marked as B. p_M, p_E and p_S are similar to p_B.

Sample Process. As the training data and testing data are sampled from different distributions, CWS models cannot learn enough features for training data. In general, if a sample sentence is more similar with sentences of the training set, the sample sentence contains more OOV words. For example, "为了 还 这份 人情 ， 冯小刚 这 才 答应 下来 。" in source domain data and "田灵儿 这 才 悻悻然 下来 。" in target domain data, "田灵儿" and "悻悻然" are both OOV words. Therefore, we select more similar target domain sentence. According to the above analysis, selecting instances relies on two aspects: The uncertainty of annotation and the semantic similarity between training data and unlabeled sentences. In general, the scoring model for selecting instances is finally defined as:

$$\mathrm{D(s)} = \lambda \sum_{i=1}^{n} E(c_i) + (1-\lambda) maxSim(s, \tilde{s}) \quad (4)$$

where, s represents a sentence in target domain data, n indicates the character number of this sentence, c_i represents the character in a sentence, $\tilde{s}$ is a training sentence, λ is a weight parameter.

3.3 Obtaining Annotated Result of Instances

To annotate instances precisely, we train an initial model on large-scale source domain data firstly, which is used to annotate instances.

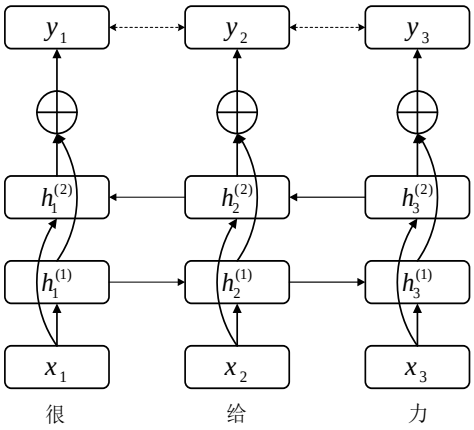

Fig. 2. Bi-LSTM-CRF neural architecture for CWS. Character vector representation are given as input; a bidirectional LSTM produces context-dependent representations; the information is passed through a hidden layer and an output layer. The outputs are confidence scores for CRFs.

Bi-LSTM-CRF Architecture for CWS. CWS task is usually solved by character-level sequence labeling algorithm. Specifically, each character in a sentence is labeled as one of $\mathrm{L} = \{\mathrm{B}, \mathrm{M}, \mathrm{E}, \mathrm{S}\}$, indicating the begin, middle, end of a word, or a word with a single character. For a given sentence $\mathrm{X} = (x_1, x_2, \ldots, x_n)$ containing n characters, the aim of CWS task is to predict label sequence $y^* = (\mathrm{y}_1, \mathrm{y}_2, \ldots, \mathrm{y}_n)$. The Bi-LSTM-CRF architecture (our baseline) for CWS is illustrated in Fig.2.

Embedding layer. Similar to other neural models, the first step is to represent characters in distributed vectors. In this work, the vector represent of a character is a concatenation of two parts: character embeddings $e^c(x_i)$ and character-bigram embeddings $e^b(x_i, x_{i+1})$. For each character x_i in a given input sentence $\mathrm{X} = (x_1, x_2, \ldots, x_n)$, we regard $e_i = [e^c(x_i), e^b(x_i, x_{i+1})]$as the vector representation of the i-th character.

Feature layers. In order to incorporate information from both sides of the sequence, we use bidirectional Long Short-Term Memory (Bi-LSTM) as feature layers. The update of each Bi-LSTM unit can be described as follows:

$$h_i = \overrightarrow{h_i} \oplus \overleftarrow{h_i} \quad (5)$$

where $\overrightarrow{h_i}$ and $\overleftarrow{h_i}$ are the hidden states at position i of the forward and backward LSTMs respectively; $\oplus$ is concatenation operation.

Inference Layer. Following Lample et al. [17], we employ conditional random fields (CRF) layer to inference labels, which is beneficial to consider the dependencies of adjacent labels. For example, a B (begin) label should be followed by a M (middle) label or E (end) label, and a M label cannot be followed by a B label or S (single) label. Given that y is a label sequence $y_1, y_2, \ldots, y_n$, then the CRF score for this sequence can be calculated as:

$$\mathrm{s(X, y)} = \sum_{i=0}^{n} A_{y_i, y_{i+1}} + \sum_{i=1}^{n} P_{i, y_i} \tag{6}$$

where A is a matrix of transition scores such that $A_{i,j}$ represents the score of a transition from the tag i to tag j. y_0 and y_n are the start and end tags of a sentence, that we add to the set of possible tags. A is therefore a square matrix of size $\mathrm{k} + 2$. P is the fractional matrix of the Bi-LSTM network's output. The size of P is $\mathrm{n} \times \mathrm{k}$, where k is the number of labels to be predicted, $P_{i,j}$ corresponds to the j-th label of the i-th word in a sentence.

The output from the model is the tagging sequence with the largest score s(y). A softmax over all possible tag sequences yields a probability for the sequence y:

$$\mathrm{p(y|X)} = \frac{e^{s(X,y)}}{\sum_{\tilde{y} \in Y_X} e^{s(X,y)}} \tag{7}$$

Train Strategy. Finally, we directly maximize the log-probability of the correct tag sequence:

$$\log\big(\mathrm{p(y|X)}\big) = \mathrm{s(X, y)} - \log\Big(\sum_{\tilde{y} \in Y_X} e^{s(X,\tilde{y})}\Big) = \mathrm{s(X, y)} - \log add_{\tilde{y} \in Y_X}(X, \tilde{y}) \tag{8}$$

While decoding, the output sequence we predict will have the highest score given by the following:

$$y^* = argmax_{\tilde{y} \in Y_X} s(X, \tilde{y}) \tag{9}$$

We use the Viterbi algorithm to find the optimal labeled sequence during training.

Revising Labeled Instances. Word segmentation systems trained on source domain often suffer a rapid decrease in performance when they are used in target domain. To mitigate the errors resulting from model segmentation scheme, we revise model segmentation results with the help of training data.

Training sentences similar to instances can be used to get rid of partial implausible annotation, the three most similar training sentences is therefore selected for every instance. We employ "Top-N" algorithm to choose them according to semantic similarity calculated in Section 3.2. Then we revise the segmentation results by comparing these three sentences with segmentation instances. Specifically, the same fragments in the three sentence and the instance are modified to be the same annotation. If the annotation of a fragment in the three sentences is inconsistent, the most similar sentence shall prevail. The rest of the instance utilize the annotation result of the model.

Table 2. Statistical information of the three datasets.

Dataset		Sentence	Words
PD	train	441943	21653284
	dev	15000	394202
	test	15000	401063
ZX	Train	2373	96934
	dev	788	20393
	test	1,394	34355
Micro-blog	train	20,135	421,166
	dev	2052	43697
	test	8,592	187,877

4 Experiments

4.1 Datasets

Source Domain Data. In this paper, we use the People's Daily[1] (PD) (2014) drawn from news domain for the source-domain training. We regard the random different 15k sentences from PD as development and test sets respectively, and the rest are treated as training sets.

Target Domain Data. The NLPCC 2016 dataset[2] [20] and "Zhuxian"[3] (hereafter referred to as ZX) are used as the target domain data. The NLPCC 2016 dataset is selected to evaluate our methods on micro-blog texts. Unlike the popular used news-wire dataset, the NLPCC 2016 dataset is collected from Sina Weibo, which consists of the informal texts from micro-blog with the various topics, such as finance, sports, entertainment, and so on. ZX is an Internet novel and has a different writing style comparing to PD. In addition, ZX also contains many novel specific named entity. Table 2 gives the details of three datasets.

Target Domain Unlabeled Data. We use micro-blog unlabeled data built from the Internet for free. After filtering special characters and removing duplication, 2.2 million sentences are reserved. For ZX, after some long sentence segmentation, we utilize about 60k sentences the rest of complete ZX novel data as ZX unlabeled data except for ZX test set.

Both recall of out-of-vocabulary words (R-oov) and F1-score are used to evaluate the segmentation performance.

4.2 Hyper-Parameter Settings

The hyper-parameters are tuned according to the experimental results. The detailed values are shown in Table 3.

[1] https://pan.baidu.com/s/1hq3KKXe
[2] https://github.com/FudanNLP/NLPCC-WordSeg-Weibo
[3] This book can be download from https://www.qisuu.la/Shtml812.html.

Table 3. Hyper-parameters Settings.

Hyper-parameter	value
Embedding dimension	100
LSTM hidden size	200
LSTM input size	200
Batch size	128
learning rate	0.01
Dropout	0.5

4.3 Experimental Results

The value of parameter λ**.** We first investigated the impact of λ in formula (9) over segmentation performance. The parameter λ is searched ranging from 0.1 to 1 with a step size of 0.1. We put λ into the formula (9) to calculate the sampling scores and select the most valuable samples. 1000 samples were selected in the experiment. The results of PD to Micro-blog and ZX datasets are shown in Fig. 3. From this figure we can see that setting λ as 0.5 gives the highest F1-score for every corpus. So the parameter λ is set to 0.5.

Effect of Samples Selection. To verify the effectiveness of our sampling strategy, we conduct a comparative experiment which choose different number of samples randomly. We apply the sampling strategy proposed in Section 3.2 to the target domain data. The annotation work of unlabeled target domain data is based on the automatically labeled sentences by our baseline model trained with PD corpus. The sentences are automatically annotated and then revised with the help of training data. Experiment results are shown in Figure 4. It can be seen that with the increase of the number of datasets, the performance becomes better. It shows that our proposed methods can learn continuously knowledge from selected instances. We can see that our method achieves better performance compared to the method with randomly target domain samples, demonstrating that sampling strategy is helpful to improving the domain adaption accuracy. And the best result (91.78%) is achieved with 32K target-domain sentences in micro-blog data set in our experiment.

In addition, to examine whether OOV recognition can benefit from our methods, we also look into the OOV recalls of the ZX dataset. Table 4 show that the proposed samples selection methods can effectively improve the recall of OOV words, which empirically proves its domain adaption ability. The main reason is our method can select sentences similar to the source domain without destroying the original distribution of the training data and alleviate the recognition problem of OOV words.

Effect of revising annotation. In order to examine the real effect of the revised methods, we also set a comparison experiment using the baseline segmentation results, as shown in the second column of Figure 5. It can be seen that the performance of revised instances is better than the unrevised instances, which indicates that our revised methods can help to correct partially incorrect word segmentation results.

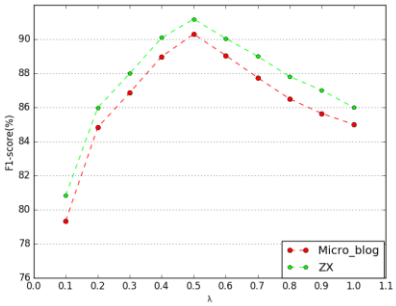

Fig. 3. The results of our methods under different parameter λ on two target domain datasets.

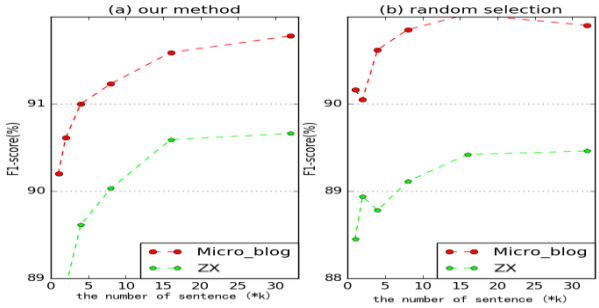

Fig. 4. F1-score on target domain data when adding different numbers of target domain data.

Table 4. The result when 16K target-domain sentences are added.

Datasets	F1-score	F1-score (unrevised)	R-oov
Micro-blog	91.78	91.40	74.62
ZX	90.66	90.35	83.56

Character-based and Word-based n-gram Embedding comparison. In addition, we add a comparative experiment by using word-based semantic similarity method. The results are shown in Table 5, where the first row shows the performance of our baseline and the second row shows the performance of the method utilizing word-based n-gram embedding, the third row shows the performance of the character-based selection method. Compared the word-based semantic similarity representations, our method performs better with character-based n-gram embedding. Similar conclusion is obtained when adapting from PD to ZX data set. It shows that character-based models have the potential of capturing morpheme patterns, thereby improving generalization ability of word segmentation models. Furthermore, F1-score in character-based experiments increases by 2.19% than our baseline.

Comparisons with State-of-the-art Models. In this section, we compare our methods with previous advanced methods. As shown in Table 6, our work achieves state-of-the-art results. Although our method is simple, it outperforms the other methods which are very competitive. Since Liu et al. [8] and Zhang et al. [9] used the domain

dictionary, our work is not comparable with them. Although contextual information is both used, our method gives better result than Bao et al. [15], showing that our method integrate context more effectively.

Table 5**.** Comparison with word-based and character-based semantic similarity.

Methods	ZX	Micro-blog
Baseline	88.47	89.90
Word-based	90.35	91.33
Character-based	**90.66**	**91.78**

Table 6. Comparisons with state-of-the-art methods on target domain datasets.

Models	ZX	Micro-blog
Liu et al. [6]	83.99	-
Qiu et al. [5]	90.2	-
Bao et al. [16]	89.35	-
Our work (words-based)	90.35	91.33
Our work(character-based)	**90.66**	**91.78**

5 Conclusion

The performance of CWS can drop significantly when the test domain is different from the training domain. In this paper, we propose a novel instances-based transfer learning method for CWS. We select useful instances containing the higher labeled values to the training set, and the labeled value is calculated by the help of character-based n-gram embedding. The model can be trained by adding training data iteratively to obtain better generalization ability. In our experiments, we evaluated two methods of semantic similarity computation: character-based and word-based. The experimental results on the Micro-blog and ZX dataset fully show that our method is especially effective for segmenting OOV words and enhancing the performance of CWS on different domains.

Acknowledgments. The authors are supported by National Nature Science Foundation of China (Contract 61370130 and 61473294), and the Fundamental Research Funds for the Central Universities(2015JBM033), and International Science and Technology Cooperation Program of China under grant No. 2014DFA11350.

References

1. Pei W., Ge T., Chang B.: Max-margin tensor neural network for Chinese word segmentation. In: The 52nd Annual Meeting of the Association for Computational Linguistics, pp. 293–303. Baltimore, Maryland (2014).

2. Chen X., Qiu X., Zhu C., Huang X.: Gated recursive neural network for Chinese word segmentation. In: The 53rd Annual Meeting of the Association for Computer Linguistics, pp. 1744–1753 (2015).
3. Cai D., Zhao H.: Neural Word Segmentation Learning for Chinese. In: The 54th Annual Meeting of the Association for Computer Linguistics, pp. 409-420 (2016).
4. Chen X., Shi Z., Qiu X., Huang X.: Adversarial Multi-Criteria Learning for Chinese Word Segmentation. pp. 1193-1203 (2017).
5. Qiu L., Zhang Y.: Word segmentation for Chinese novels. In: AAAI. pp. 2440–2446 (2015).
6. Liu Y., Zhang Y.: Unsupervised domain adaptation for joint segmentation and POS-tagging. In: Proceedings of COLING 2012: Posters. pp. 745–754. The COLING 2012 Organizing Committee (2012).
7. Daume H. and Marcu D.: Domain adaptation for statistical classifiers. In: Journal of Artificial Intelligence Research, pp. 101–126 (2006).
8. Liu Y., Zhang Y., Che W., Liu T., Wu F.: Domain adaptation for CRF-based Chinese word segmentation using free annotations. In: EMNLP (2014).
9. Zhang M., Zhang Y., Che W., Liu T.: Type-supervised domain adaptation for joint segmentation and pos-tagging. In: EACL, pp. 588–597(2014).
10. W J., L H., Q L., Lü Y.: A Cascaded Linear Model for Joint Chinese Word Segmentation and Part-of-Speech Tagging. In: Meeting of the Association for Computational Linguistics, pp. 897-904. June 15-20, 2008, Columbus, Ohio, Usa. DBLP (2008).
11. Xu J., Ma S., Zhang Y., Wei B., Cai X., Sun X.: Transfer Deep Learning for Low-Resource Chinese Word Segmentation with a Novel Neural Network. pp. 721-730 (2017).
12. Blitzer J., McDonald R., Pereira F.: Domain adaptation with structural correspondence learning. In: EMNLP, pp. 120–128 (2006).
13. Choi H., Cho K., Bengio Y.: Context-dependent word representation for neural machine translation. In: Computer Speech and Language (2016).
14. Qin L., Zhang Z., Zhao H.: Implicit discourse relation recognition with context-aware character-enhanced embeddings. In: The 26th International Conference on Computational Linguistics (COLING), Osaka, Japan, December (2016)
15. Bao Z., Li S., Gao S., Xu W.: Neural Domain Adaptation with Contextualized Character Embedding for Chinese Word Segmentation (2017).
16. Zhou H., Yu Z., Zhang Y., Huang S., Dai X.: Word-Context Character Embeddings for Chinese Word Segmentation. In: Conference on Empirical Methods in Natural Language Processing. pp. 760-766(2017).
17. Lample G., Ballesteros M., Subramanian S., Kawakami K., Dyer C.: Neural Architectures for Named Entity Recognition. pp. 260-270 (2016).
18. Mikolov T., Chen K., Corrado G., Dean J.: Efficient Estimation of Word Representations in Vector Space. Computer Science (2013).
19. Zheng X., Chen H., Xu T.: Deep learning for Chinese word segmentation and POS tagging. In: Proceedings of the 2013 Conference on Empirical Methods in Natural Language Processing. pp. 647–657. Association for Computational Linguistics (2013).
20. Qiu X., Qian P., Shi Z.: Overview of the NLPCC-ICCPOL 2016 shared task: Chinese word segmentation for micro-blog texts. In: The International Conference on Computer Processing of Oriental Languages. Springer, pp. 901–906 (2016).
21. Lin D., An X., Zhang J.: Double-bootstrapping source data selection for instance-based transfer learning. Pattern Recognition Letters 34(11), 1279-1285 (2013).