Proceedings of the Twenty-Seventh International Joint Conference on Artificial Intelligence (IJCAI-18)

Neural Networks Incorporating Unlabeled and Partially-labeled Data for Cross-domain Chinese Word Segmentation

Lujun Zhao, Qi Zhang, Peng Wang, Xiaoyu Liu
School of Computer Science, Fudan University, Shanghai, China
Shanghai Key Laboratory of Intelligent Information Processing, Shanghai, China
{ljzhao16,qz,wangp15,liuxiaoyu16}@fudan.edu.cn

Abstract

Most existing Chinese word segmentation (CWS) methods are usually supervised. Hence, large-scale annotated domain-specific datasets are needed for training. In this paper, we seek to address the problem of CWS for the resource-poor domains that lack annotated data. A novel neural network model is proposed to incorporate unlabeled and partially-labeled data. To make use of unlabeled data, we combine a bidirectional LSTM segmentation model with two character-level language models using a gate mechanism. These language models can capture co-occurrence information. To make use of partially-labeled data, we modify the original cross entropy loss function of RNN. Experimental results demonstrate that the method performs well on CWS tasks in a series of domains.

1 Introduction

In contrast to Western languages, Chinese has no explicit word delimiters in a sequence of text. Hence, Chinese language processing usually require Chinese word segmentation (CWS) as a pre-processing step. CWS methods usually treat the task as a sequential labeling problem. For a given sentence, labels are assigned to all of the characters. These tags may indicate the position of a character in the word [Xue, 2003] or represent the intervals between characters [Huang *et al.*, 2007]. Recently, along with the development of deep learning methods, some neural network models [Chen *et al.*, 2015; Cai and Zhao, 2016; Zhang *et al.*, 2016; Liu *et al.*, 2016; Cai *et al.*, 2017] have achieved great success in CWS tasks. Despite their enormous success, however, these methods still have limitations: they usually rely heavily on manually labeled training data. These data mostly comes from the newswire domain. From previous studies [Liu and Zhang, 2012], we know that when the evaluation data shift from the newswire domain to other domain, the performance of supervised methods usually drops severely.

Although annotated domain-specific datasets cannot be easily achieved and manual annotation is a time consuming process, we can easily find large-scale unlabeled data over the Internet. Many works have investigated the use of such free data to address the problems for the resource-poor domains [Jin and Tanaka-Ishii, 2006; Zhao and Kit, 2008; Li and Sun, 2009; Sun and Xu, 2011; Zhang *et al.*, 2013]. Meanwhile, from web pages such as Wikipedia, a sentence's partial segmentation information can be inferred from hyperlinks, which produces partially-labeled data. In addition, with the help of lexicons, we can also obtain partially-labeled data by matching characters. Previous works have shown that this can provide valuable information for cross-domain CWS tasks [Jiang *et al.*, 2013; Liu *et al.*, 2014]. Although these approaches have demonstrated clear benefits of incorporating unlabeled and partially-labeled data, how to integrate such data into neural network models for CWS tasks in an end-to-end manner is still not well investigated.

In this work, we propose a neural sequence labeling architecture to utilize unlabeled and partially-labeled data for cross-domain CWS tasks. By definition, a word is composed of characters. Thus, if two characters form a word (or a part of a word), they will frequently occur together in succession. This co-occurrence information, which can be obtained from unlabeled data, can benefit the CWS tasks. To utilize this co-occurrence information, we combine the segmentation model, implemented by BiLSTM, with two character-level language models. Such language models can capture the character-level co-occurrence information. In addition, we use a gate mechanism to learn the degree of influence exerted by the co-occurrence information on the segmentation model. To handle partially-labeled data, we modify the objective function used in the training of sequence labeling RNN, which allows the proposed method to be trained with either full or partial labels. Experimental results show that the proposed method could easily shift between different domains and performed well on cross-domain CWS tasks.

Our contributions are as follows: 1) We introduce a method to incorporate unlabeled data, which combines the segmentation model with language models. A gate mechanism is used to learn the degree of influence exerted by the language models. 2) We modify the loss function used in CWS tasks to handle the partially-labeled data. 3) Based on several experiments, we demonstrate that the proposed method can achieve better performance than previous state-of-the-art methods.

2 Neural Model for CWS

The common choice for sequence labeling CWS is the BMES scheme, which is to label each character in a sentence as one of $\{B, M, E, S\}$, representing the beginning, middle, or end of a word, or a word with a single character, respectively. We adopt recurrent neural network with long short-term memory (LSTM) as our segmentation model because it is capable of reducing the feature engineering [Chen *et al.*, 2015].

Long Short-term Memory LSTM [Hochreiter and Schmidhuber, 1997] is a type of RNN designed to cope with gradient vanishing problems. Basically, the formulas to update an LSTM unit at time t can be abbreviated as follows:

$$\boldsymbol{h}_t = \text{LSTM}(\boldsymbol{x}_t, \boldsymbol{h}_{t-1}; \boldsymbol{\theta}). \tag{1}$$

Here, $\boldsymbol{x}_t$ is the input vector and $\boldsymbol{h}_t$ is the hidden state; $\boldsymbol{\theta}$ represents all the parameters of the LSTM.

In CWS tasks, it is beneficial to access both past and future contexts. Thus, we use a bidirectional LSTM (BiLSTM). The update of the BiLSTM unit can be written as follows:

$$\boldsymbol{h}_t = [\overrightarrow{\boldsymbol{h}}_t; \overleftarrow{\boldsymbol{h}}_t] = \text{BiLSTM}(\boldsymbol{x}_t, \overrightarrow{\boldsymbol{h}}_{t-1}, \overleftarrow{\boldsymbol{h}}_{t+1}; \boldsymbol{\theta}_{bi}), \tag{2}$$

where $\overrightarrow{\boldsymbol{h}}_t$ is the forward hidden state and $\overleftarrow{\boldsymbol{h}}_t$ is the backward hidden state at time t; $\boldsymbol{h}_t$ is the concatenation of $\overrightarrow{\boldsymbol{h}}_t$ and $\overleftarrow{\boldsymbol{h}}_t$; and $\boldsymbol{\theta}_{bi}$ represents all the parameters of the BiLSTM.

CWS with BiLSTM Given a sentence S represented by a sequence of characters $[c_1, c_2, \ldots, c_T]$, we first use an embedding layer to obtain the vector representation (embeddings) $\boldsymbol{x}_t$ for each character c_t. Then, $\boldsymbol{x}_t$ is fed to the BiLSTM. At time t, $\boldsymbol{h}_t$ in equation (2) can be regarded as the representation of c_t, which incorporates the context information. Through a softmax non-linear layer, the output $\hat{\boldsymbol{y}}_t$ can be obtained by

$$\hat{\boldsymbol{y}}_t = \text{softmax}(\boldsymbol{W}\boldsymbol{h}_t + \boldsymbol{b}), \tag{3}$$

where $\hat{\boldsymbol{y}}_t$ represents the prediction probabilities, $\boldsymbol{W}$ is the weight matrix, and $\boldsymbol{b}$ is the bias.

Given the ground-truth labels $\boldsymbol{y}_1, \boldsymbol{y}_2, \ldots, \boldsymbol{y}_T$ represented by one-hot vector, the cross entropy loss function for sentence S is given by the following:

$$\mathcal{L}(\boldsymbol{Y}, \hat{\boldsymbol{Y}}) = -\frac{1}{T}\sum_{t=1}^{T} \boldsymbol{y}_t^\top \log \hat{\boldsymbol{y}}_t, \tag{4}$$

where $\boldsymbol{Y} = \{\boldsymbol{y}_1, \boldsymbol{y}_2, \ldots, \boldsymbol{y}_T\}$ and $\hat{\boldsymbol{Y}} = \{\hat{\boldsymbol{y}}_1, \hat{\boldsymbol{y}}_2, \ldots, \hat{\boldsymbol{y}}_T\}$.

A CRF layer can be further used to leverage the dependencies of adjacent labels [Huang *et al.*, 2015].

3 Incorporating Unlabeled Data for Cross-domain CWS

The above BiLSTM segmentation model does not work well when the training and test datasets come from different domain (namely source domain and target domain). It is unrealistic to annotate a dataset of target domain for training. However, there is a large of unlabeled data on the Internet. This leads to the unsupervised domain adaptation problem. In this setting, for the source domain, we have a labeled dataset $\mathcal{S}_l$, while for the target domain, we only have an unlabeled dataset $\mathcal{T}_u$. Using $\mathcal{S}_l$ and $\mathcal{T}_u$, our goal is to train a model with good performance in the target domain at the evaluation time.

3.1 Motivation

Because a word is composed of characters, two characters will frequently occur together in succession if they form a word (or a part of a word). This co-occurrence information, which can be obtained from unlabeled data, can benefit the CWS tasks. To utilize this information, we propose to combine character-level language model with the BiLSTM segmentation model. The objective of language model is to predict the next character c_t, given the previous $t-1$ characters $c_1, c_2, \ldots, c_{t-1}$. To see why language model can incorporate co-occurrence information, here is an example in English. Given a sequence of characters "American presi", it is easy to predict that the next characters are "dent" because we know "president" is a word and "presi" and "dent" co-occur many times. However, it is hard to predict that the next characters are "presi" given "American" because they belong to different words. Similarly in Chinese, we find that, in the sequence "美国|总统" (美国(American) is a word and 总统(president) is another word), the probability p("统"|"美国总") given by language model is 0.71, but the probability p("总"|"美国") is 0.05[1]. In short, when language model gives a high probability $p(c_t|c_1, c_2, \ldots, c_{t-1})$, it tells us that c_t and c_{t-1} should not be segmented, but when language model gives a low probability, it tells us that c_t and c_{t-1} should be segmented.

3.2 Network Architecture

The architecture of our proposed model is illustrated in Figure 1. It mainly includes three components: the *forward language model* (pink), *backward language model* (yellow), and *BiLSTM segmentation model* (blue). The forward and backward language models are also implemented by LSTM.

Forward Language Model In the forward language model, we feed the embeddings $\{\boldsymbol{x}_1, \boldsymbol{x}_2, \ldots, \boldsymbol{x}_T\}$ of characters $\{c_1, c_2, \ldots, c_T\}$ sequentially, one at a time. At each time-step t, we want to predict the next character c_{t+1} based on the information incorporated from $c_1, c_2, \ldots, c_t$, which is denoted by $\boldsymbol{h}_t^f$. Concretely, the formulas at time t are as follows:

$$\boldsymbol{h}_t^f = \text{LSTM}^f(\boldsymbol{x}_t, \boldsymbol{h}_{t-1}^f; \boldsymbol{\theta}_f), \tag{5}$$

where $\boldsymbol{\theta}_f$ denotes all the parameters of LSTM^f. Then, a following softmax non-linear layer outputs the probability distribution over the vocabulary:

$$\hat{\boldsymbol{y}}_t^f = \text{softmax}(\boldsymbol{W}_f\boldsymbol{h}_t^f + \boldsymbol{b}_f), \tag{6}$$

where $\hat{\boldsymbol{y}}_t^f$ represents the prediction probabilities of next character c_{t+1}, $\boldsymbol{W}_f$ is the weight matrix, and $\boldsymbol{b}_f$ is the bias.

Backward Language Model The basic structure of the backward language model is the same as that of the forward language model, with the difference that we feed the embeddings $\{\boldsymbol{x}_1, \boldsymbol{x}_2, \ldots, \boldsymbol{x}_T\}$ in reverse order, i.e., beginning with $\boldsymbol{x}_T$. Then, we feed $\boldsymbol{x}_{T-1}, \boldsymbol{x}_{T-2}, \ldots$ and end with $\boldsymbol{x}_1$. At each time-step, we want to predict the previous character c_{t-1} based on the information incorporated from $c_T, c_{T-1}, \ldots, c_t$, which is denoted by $\boldsymbol{h}_t^b$:

$$\boldsymbol{h}_t^b = \text{LSTM}^b(\boldsymbol{x}_t, \boldsymbol{h}_{t+1}^b; \boldsymbol{\theta}_b), \tag{7}$$

[1]The results are obtained by training a LSTM-based language model on People's Daily dataset.

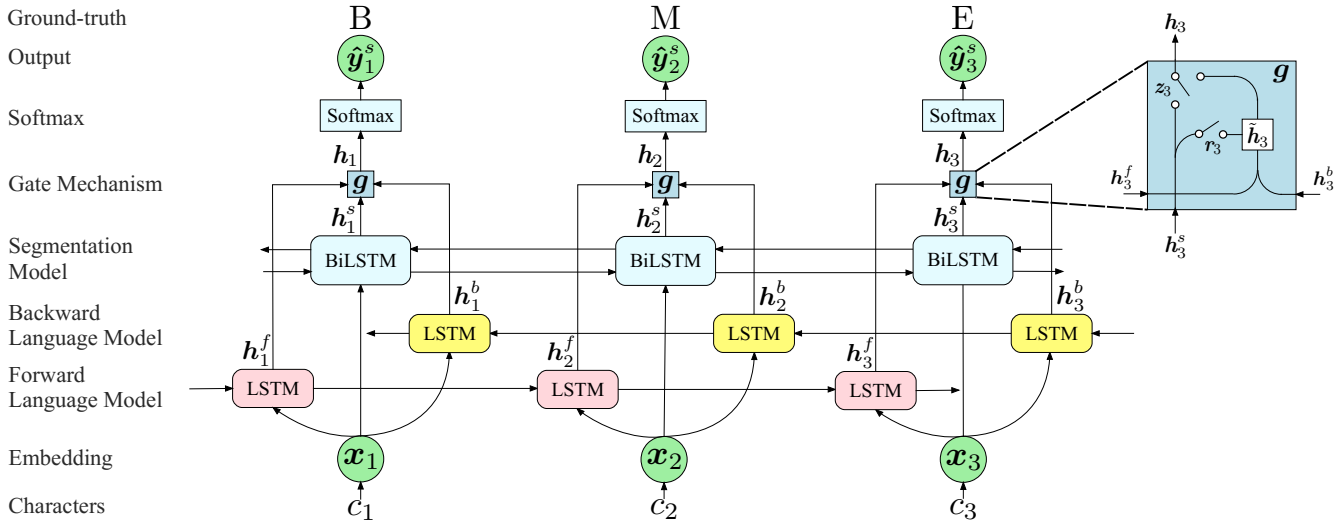

Figure 1: Architecture of our proposed model. It mainly includes three components: the *forward language model* (pink), *backward language model* (yellow), and *BiLSTM segmentation model* (blue). We use a gate mechanism to control the influence of the language models on the segmentation model. The outputs of language models are not shown for simplicity. In this example, we assume that "$c_1c_2c_3$" is a word.

where $\boldsymbol{\theta}_b$ denotes all the parameters of LSTM^b. Then, the output of softmax is as follows:

$$\hat{\boldsymbol{y}}_t^b = \text{softmax}(\boldsymbol{W}_b \boldsymbol{h}_t^b + \boldsymbol{b}_b), \tag{8}$$

where $\hat{\boldsymbol{y}}_t^b$ represents the prediction probabilities of previous character c_{t-1}, $\boldsymbol{W}_b$ is the weight matrix, and $\boldsymbol{b}_b$ is the bias.

Segmentation Model In the segmentation model, we still use BiLSTM. The update at time t can be written as follows:

$$\boldsymbol{h}_t^s = [\overrightarrow{\boldsymbol{h}}_t^s; \overleftarrow{\boldsymbol{h}}_t^s] = \text{BiLSTM}^s(\boldsymbol{x}_t, \overrightarrow{\boldsymbol{h}}_{t-1}^s, \overleftarrow{\boldsymbol{h}}_{t+1}^s; \boldsymbol{\theta}_s), \tag{9}$$

where $\boldsymbol{\theta}_s$ denotes all the parameters of BiLSTM^s.

Gate Mechanism To learn the degree of influence of co-occurrence information on the segmentation model, we use a gate mechanism similar to a GRU [Cho *et al.*, 2014]:

$$\begin{aligned}
\boldsymbol{z}_t &= \boldsymbol{\sigma}(\boldsymbol{U}_z \boldsymbol{h}_t^f + \boldsymbol{V}_z \boldsymbol{h}_t^b + \boldsymbol{W}_z \boldsymbol{h}_t^s + \boldsymbol{b}_z) \\
\boldsymbol{r}_t &= \boldsymbol{\sigma}(\boldsymbol{U}_r \boldsymbol{h}_t^f + \boldsymbol{V}_r \boldsymbol{h}_t^b + \boldsymbol{W}_r \boldsymbol{h}_t^s + \boldsymbol{b}_r) \\
\tilde{\boldsymbol{h}}_t &= \textbf{tanh}(\boldsymbol{U}_h \boldsymbol{h}_t^f + \boldsymbol{V}_h \boldsymbol{h}_t^b + \boldsymbol{W}_h(\boldsymbol{r}_t \odot \boldsymbol{h}_t^s) + \boldsymbol{b}_h) \\
\boldsymbol{h}_t &= (1 - \boldsymbol{z}_t) \odot \boldsymbol{h}_t^s + \boldsymbol{z}_t \odot \tilde{\boldsymbol{h}}_t
\end{aligned} \tag{10}$$

where $\boldsymbol{z}_t$ and $\boldsymbol{r}_t$ are the update and reset gate respectively. The new hidden state $\boldsymbol{h}_t$ is connected to a dense layer with linear transformation and softmax output to predict the segmentation label:

$$\hat{\boldsymbol{y}}_t^s = \text{softmax}(\boldsymbol{W} \boldsymbol{h}_t + \boldsymbol{b}), \tag{11}$$

where $\hat{\boldsymbol{y}}_t^s$ is the probability distribution over all the segmentation labels $\{B, M, E, S\}$.

3.3 Optimization

The training set consists of two parts: the labeled source domain dataset $\mathcal{S}_l$, of which we know the segmentation label and unlabeled target domain dataset $\mathcal{T}_u$, of which we only have the raw text. Hence, for dataset $\mathcal{S}_l$, we optimize the two language models and segmentation model together using the standard cross-entropy loss, but for dataset $\mathcal{T}_u$, we simply optimize the two language models.

4 Incorporating Partially-labeled Data for Cross-domain CWS

4.1 Partially-labeled Data

We assume there is also a partially-labeled dataset $\mathcal{T}_p$ besides the unlabeled dataset $\mathcal{T}_u$ for the target domain. We consider a weakly-supervised sequence labeling setting where each sequence is partially-labeled: only a few members are labeled or we are given a candidate set of labels, only one of which is correct. Liu et al. [2014] proposed two methods to obtain partially-labeled data: lexicons and natural annotation. We also use these in this work. To use a lexicon, we match the words in the lexicon with unlabeled data, resulting in partially-labeled sentences. To use natural annotation, we culled a large amount of web text (such as from Wikipedia) from the Internet, and used this web text (usually containing many hyperlinks and other markup annotations) to obtain partially-labeled sentences. Figure 2 illustrates an example of partially-labeled data. In the sentence "美国总统住在白宫 (American president lives in the White House)", we only know that "总统 (president)" is a word (probably inferred from a lexicon or hyperlink). Hence, the labels of "总" and "统" are "B" and "E", respectively. Because only "E", "S" can be followed by "B" and only "B", "S" can follow "E", we infer that the candidate sets of labels for "国" and "住" are $\{E, S\}$ and $\{B, S\}$, respectively.

4.2 Training RNN with Partially-labeled Data

In the CWS tasks, there are only four segmentation labels: B, M, E, S. We use a zero-one vector to represent these labels. For example, "B" is represented as $\boldsymbol{y} = [1, 0, 0, 0]^\top$ (also known as one-hot vector) and a candidate set of labels like $\{E, S\}$ is represented as $\boldsymbol{y}^c = [0, 0, 1, 1]^\top$ (we add a superscript to distinguish it (candidate set) from the one-hot vector (single label)).

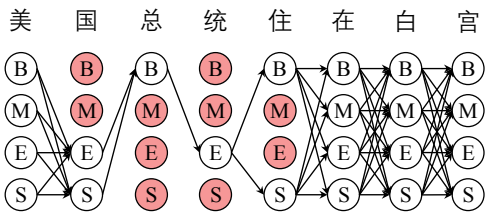

Figure 2: Examples of partially-labeled data. In the sentence "美国总统住在白宫 (American president lives in the White House)", we only know "总统 (president)" is a word. The paths show possible correct segmentations.

In a vanilla RNN for a sequence labeling problem, given an input sequence $\boldsymbol{x}_1, \boldsymbol{x}_2, \ldots, \boldsymbol{x}_T$, we know the correct label $\boldsymbol{y}_t$ for each $\boldsymbol{x}_t$. Let $\hat{\boldsymbol{y}}_1, \hat{\boldsymbol{y}}_2, \ldots, \hat{\boldsymbol{y}}_T$ be the prediction probabilities given by this RNN. To train this network, a common choice for the loss function is the cross-entropy loss function:

$$\mathcal{L}(\boldsymbol{Y}, \hat{\boldsymbol{Y}}) = -\frac{1}{T}\sum_{t=1}^{T} \boldsymbol{y}_t^\top \log \hat{\boldsymbol{y}}_t, \tag{12}$$

where $\boldsymbol{Y} = \{\boldsymbol{y}_1, \boldsymbol{y}_2, \ldots, \boldsymbol{y}_T\}$ and $\hat{\boldsymbol{Y}} = \{\hat{\boldsymbol{y}}_1, \hat{\boldsymbol{y}}_2, \ldots, \hat{\boldsymbol{y}}_T\}$.

For a sequence $\boldsymbol{x}_1, \boldsymbol{x}_2, \ldots, \boldsymbol{x}_T$ which is partially-labeled, only a few inputs $\boldsymbol{x}_t$ have a corresponding label or a candidate set of labels, while many others do not. Thus, the loss function (12) cannot be used for training over partially-labeled data. Let $\mathcal{I}$ denote the set of indices: for each $i \in \mathcal{I}$, $\boldsymbol{x}_i$ is given a correct label $\boldsymbol{y}_i$ and let $\mathcal{J}$ denote another set of indices: for each $j \in \mathcal{J}$, $\boldsymbol{x}_j$ is given a candidate set of labels $\boldsymbol{y}_j^c$. We optimize the following loss function $\mathcal{L}_p$ for partially-labeled data:

$$\mathcal{L}_p = \underbrace{-\frac{1}{|\mathcal{I}|}\sum_{i\in\mathcal{I}} \boldsymbol{y}_i^\top \log \hat{\boldsymbol{y}}_i}_{\text{first term}} \underbrace{-\frac{1}{||\mathcal{J}||}\sum_{j\in\mathcal{J}} (\mathbf{1} - \boldsymbol{y}_j^c)^\top \log(\mathbf{1} - \hat{\boldsymbol{y}}_j)}_{\text{second term}}, \tag{13}$$

where $|\mathcal{I}|$ denotes the size of $\mathcal{I}$ and we define $||\mathcal{J}||$ as $\sum_{j\in\mathcal{J}} ||\mathbf{1} - \boldsymbol{y}_j^c||_0$. $||\cdot||_0$ is the l_0-norm. The first term is easy to understand: we minimize the cross-entropy loss on $\mathcal{I}$, which actually maximizes the prediction probability of the correct label. Here is our motivation for understanding the second term. Given a candidate set of labels, we do not know which one is the correct label, so we cannot maximize its prediction probability of it. We know, however, which labels are the incorrect labels. For example, the candidate set is $\{E, S\}$, so "B" and "M" are the incorrect labels. Instead of maximizing the prediction probability of the correct label, we can minimize the prediction probability of the incorrect labels which leads to the second term of equation (13).

The function $\mathcal{L}$ in equation (12) can be regarded as a special case of $\mathcal{L}_p$, where $\mathcal{I} = \{1, 2, \ldots, T\}$.

5 Experiments

5.1 Datasets

To evaluate the proposed model, we performed cross-domain experiments on various domain datasets, following the previous works of Liu et al. [2014] and Zhang et al. [2014]. For the source domain, we used the labeled People's Daily (PD) dataset. For the target domain, we used four datasets from the finance, medicine, literature, and computer fields. These test and development data could be obtained from SIGHAN Bakeoff 2010. The unlabeled data for the target domain were culled from the Internet[2] (about 200K sentences for each domain), and the partially-labeled data were collected from Chinese Wikipedia[3] (about 100K sentences for each domain).

Another experiment was also conducted, using Chinese Treebank 5.0 (CTB5) as the source domain and using Zhuxian (a Chinese Internet novel, also known as the "Jade dynasty") dataset[4] (ZX) as the target domain. The ZX dataset was annotated by Liu and Zhang [2012] and Zhang et al. [2014], and the style is very different from CTB5. We collected the remaining part of this novel as our unlabeled data (about 26K). To obtain partially-labeled data (about 24K), we used a lexicon containing 51 names of characters, following Zhang et al. [2014].

All of the datasets were preprocessed using regular expressions, and the continuous English characters and digits were marked as not segmented.

5.2 Setups

In this work, we trained a total of five different models:

- **BiLSTM**. Our baseline model. In BiLSTM, we used a two-layer bi-directional LSTM. In the training stage, we only used the labeled source domain dataset $\mathcal{S}_l$.
- **BiLSTM + CRF**. Another baseline model which incorporates CRF in the output layer.
- **BiLSTM + PL**. Besides using dataset $\mathcal{S}_l$, we also used the partially-labeled target domain dataset $\mathcal{T}_p$ to train the BiLSTM segmentation model.
- **BiLSTM + LM**. In addition to BiLSTM segmentation model, we added two extra language models to incorporate co-occurrence information using unlabeled data. In both language models, we used a two-layer LSTM. In the training stage, we used the labeled source domain dataset $\mathcal{S}_l$ and the unlabeled target domain dataset $\mathcal{T}_u$.
- **BiLSTM + LM + PL**. On the basis of the previous model, we also used the partially-labeled target domain dataset $\mathcal{T}_p$ in the training stage.

For all five models, we used Adam [Kingma and Ba, 2014] to train our networks, and the initial learning rate was set at 0.001. Every five epochs, the learning rate decayed by half. In order to avoid overfitting, we employed the dropout technique (20% dropout rate). We performed gradient clipping (we clipped the norm of the gradients at 5) to cope with the exploding gradient problem. The character embeddings were initialized by pre-trained embedding based on word2vec and then fine-tuned during training. We set the character embedding size to 100 and the LSTM hidden state size to 300. We

[2] http://www.cnki.net/.

[3] We used wikidump20161201 downloaded from https://dumps.wikimedia.org/zhwiki/20161201/.

[4] These data are available in http://zhangmeishan.github.io/eacl14mszhang.zip.

Model / Source / Target	PD Finance	PD Medicine	PD Literature	PD Computer	CTB5 ZX	Avg.
[Chen *et al.*, 2015]	95.20	92.16	92.89	93.71	87.56	92.30
[Cai and Zhao, 2016]	95.53	91.13	92.87	94.12	87.40	92.21
[Cai *et al.*, 2017]	95.38	92.10	92.90	94.04	87.96	92.48
[Zhang *et al.*, 2014]	-	-	-	-	88.34	88.34
[Liu *et al.*, 2014]	95.54	92.63	92.49	94.07	90.63	93.07
[Jiang *et al.*, 2013]	93.16	93.34	93.53	91.19	-	92.81
[Zhou *et al.*, 2017]	-	-	-	-	90.10	90.10
[Huang *et al.*, 2017]	95.81	92.26	**94.33**	93.99	-	94.10
BiLSTM	95.19	91.72	92.60	94.35	89.05	92.58
BiLSTM + CRF	95.59	92.36	93.32	94.56	89.44	93.05
BiLSTM + PL	95.25	93.20	92.31	94.78	91.64	93.43
BiLSTM + LM	95.71	93.01	93.58	95.09	90.60	93.60
BiLSTM + LM + PL	**95.84**	**93.73**	93.23	**95.32**	**92.86**	**94.20**

Table 1: Comparison with previous models using F1 scores. The first three rows are supervised methods and the middle five rows show the results of other domain adaptation methods. The last five rows shows the two baseline models and our three models.

also used bigram character embedding for the two baseline models, but we did not use it in our three models since the number of bigram character is very large (we use many unlabeled and partial labeled data) and it is very easy to overfitting. We used early stopping based on the performance on the development set. Our code will be made publicly available for reproducibility.

5.3 Results and Discussion

We first compare our methods with three supervised neural network methods, which are generally applied to domain-specific datasets. The method of Chen et al. [2015] is a character-based method and that of Cai and Zhao [2016] is a word-based method. The method of Cai et al. [2017] is a new improved version of that by Cai and Zhao [2016]. The three methods do not use any unlabeled or partially-labeled data except source domain labeled data $\mathcal{S}_l$. The results (F1 scores) are shown in the first three rows of Table 1.

We also compared our methods with five other domain adaptation methods. The method of Zhang et al. [2014] is a domain adaptation method for joint Chinese segmentation and POS-tagging, which uses an external dictionary to improve its performance. They only presented the result on ZX test data, and we took it directly from their paper. Liu et al. [2014] adopted a CRF-based method, which exploits two sources of partially annotated data: lexicons and natural annotation. Jiang et al. [2013] also used massive natural annotations. Zhou et al. [2017] proposeed word-context character embeddings for semi-supervised CWS. Huang et al. [2017] incorporated a global recurrent structure to model boundary features dynamically. The results of these methods are listed in the middle five rows of Table 1.

The last five rows of Table 1 list the results of two baseline models and our three models. The results demonstrated that compared with the baseline model "BiLSTM", the model "BiLSTM + LM" incorporating unlabeled data and the model "BiLSTM + PL" incorporating partially-labeled data achieved better performances. Literature, however, was an exception. The F1 of model "BiLSTM + PL" fell to 92.31, compared with 92.60 for "BiLSTM". We suppose that there were two reasons for this. First, the words (e.g., idioms) used in Literature are not partially-labeled in Wikipedia. Second, the noise in Wikipedia has a bad effect on the performance. For example, the sentence "他|下意识|地|伸手" (he stretched out his hand subconsciously) is mis-segmented by "BiLSTM+PL" as "他|下|意识|地|伸手" and a Chinese idiom "高楼大厦" (which means "tall buildings") is mis-segmented as "高楼|大厦" because "意识" (consciousness) and "大厦" (skyscraper) are partially-labeled in Wikipedia. The BiLSTM model can give the correct segmentation for these two examples. In the end, when we used unlabeled and partially-labeled data together, our method (BiLSTM + LM + PL) obtained a state-of-the-art performance for almost all of the test sets except Literature.

Model	Fin	Med	Lit	Com	ZX
BiLSTM	70.67	66.06	60.62	70.30	74.00
+ PL	75.00	76.39	63.46	79.96	84.03
+ LM	75.25	71.43	64.42	75.07	77.82
+ LM + PL	**76.42**	**78.11**	**66.25**	**81.39**	**86.27**

Table 2: Comparison using OOV recall when we add LM or PL.

Gate	Fin	Med	Lit	Com	ZX
No	95.48	92.40	93.21	94.44	90.42
Yes	**95.71**	**93.01**	**93.58**	**95.09**	**90.60**

Table 3: Results (F1 score) with and without gate mechanism in "BiLSTM + LM".

5.4 Analysis

Improvement in OOV recall. Out of vocabulary (OOV) words are the main challenge in cross-domain CWS tasks.

Model	MSR	PKU	CTB6
BiLSTM	96.50	95.05	95.16
+ PL	96.56	95.17	95.54
+ LM	96.84	95.39	95.57
+ LM + PL	**96.89**	**95.55**	**95.94**

Table 4: Results (F1 score) on the three standard benchmarks.

In this experiment, we studied the improvement in the OOV recall brought by incorporating unlabeled or partially-labeled data. Table 2 shows the results of our models compared with BiLSTM model for five datasets: Finance (Fin), Medicine (Med), Literature (Lit), Computer (Com), and ZX. We can observed that the OOV recall can be significantly improved.

Importance of gate mechanism. To understand the importance of the gate mechanism introduced in the "BiLSTM + LM" model, we ran an experiment that removed the gate. We used a concatenation of $\boldsymbol{h}_t^f, \boldsymbol{h}_t^b$ and $\boldsymbol{h}_t^s$ to obtain the the new hidden state $\boldsymbol{h}_t$ that replaced the gate mechanism in equation (10). Results in Table 3 showed that without gate mechanism, the performance declined in all of five datasets.

Performance with in-domain data. Although the goal of this study was to tackle the cross-domain CWS, we also wanted to know whether these methods worked for in-domain data (i.e., where the test data come from the same domain as the training data). For this reason, we performed experiments on standard benchmarks (MSR, PKU, CTB6). We used 10% data of the train set as the development set for all datasets. Since all of the three datasets were from the newswire domain, when we performed experiment on one of the datasets, we used another two datasets as unlabeled data. The partially-labeled data was also collected from Wikipedia. The results are shown in Table 4. We can find that compared with BiLSTM, incorporating unlabeled and partially-labeled data also can improve the performance of in-domain data.

Impact of unlabeled and partially-labeled data. In our methods, we used a large amount of unlabeled and partially-labeled data to improve the performance on the target domain. To investigate the impact of these data, we performed experiments on various sizes ($3\%, 16\%, 33\%, 66\%, 100\%$) of unlabeled and partially-labeled data. The results with the "BiLSTM + LM" model when using various sizes of unlabeled data are shown in Table 5, and the results with the "BiLSTM + PL" model when using various sizes of partially-labeled data are shown in Table 6. From these tables, we can draw two conclusions: (1) With an increase in the amount of unlabeled data, the F1 score gradually grows as well, demonstrating that valuable information can indeed be extracted from the unlabeled data. (2) The situation is somewhat different for the partially-labeled data. the F1 score first rises up and then moves down, showing that the harmful noise contained in partially-labeled data has a negative effects on the model when the amount of data is large enough.

Percent	Fin	Med	Lit	Com	ZX
3%	95.46	92.63	92.84	94.55	89.60
16%	95.57	92.92	93.15	94.63	90.07
33%	95.66	92.96	93.29	94.68	90.42
66%	95.68	93.00	93.43	95.03	90.59
100%	**95.71**	**93.01**	**93.58**	**95.09**	**90.60**

Table 5: F1 score of model "BiLSTM + LM" when using various sizes of unlabeled data.

Percent	Fin	Med	Lit	Com	ZX
3%	95.30	92.53	**92.66**	94.64	91.67
16%	95.37	92.87	92.59	94.77	91.92
33%	**95.43**	93.11	92.38	94.89	92.07
66%	95.30	**93.22**	92.32	**94.95**	**92.09**
100%	95.25	93.20	92.31	94.78	91.64

Table 6: F1 score of model "BiLSTM + PL" when using various sizes of partially-labeled data.

6 Related Work

Use of Unlabeled Data. Jin and TanakaIshii [2006] used branching entropy to detect boundary between characters; Li and Sun [2009] and Zhang et al. [2013] incorporated punctuation information; Sun and Xu [2011] exploited statistics-based features distilled from unlabeled data; Shen et al. [2013] extracted maximized substrings from unlabeled data and Yang et al. [2017] used unlabeled data to pre-train submodule. Different from these models, our method utilize unsupervised co-occurrence information provided by language models. Recently, Rei [2017] and Peters et al. [2017] also proposed using language models in semi-supervised frameworks. Different from us, the method of Rei [2017] is a multitask learning method, in which the language models and the sequence labeling model share a bidirectional LSTM. The method of Peters et al. [2017] was inspired by the pre-trained word embeddings. They proposed using pre-trained bidirectional LM to add context embeddings in sequence labeling.

Use of Partially-labeled Data. Jiang et al. [2013] proposed a novel discriminative learning algorithm to utilize the knowledge in the massive natural annotations on the Internet for Chinese word segmentation. Tsuboi et al. [2008] and Triggs and Verbeek [2008] independently proposed the parameter estimation method for CRFs using partially-labeled data by marginalizing out the unknown labels. Thereafter, Yang and Vozila [2014] and Liu et al. [2014] applied this method to Chinese word segmentation. They also make use of Wikipedia data as partially-labeled data. In contrast, we proposed a method to train RNN with partially-labeled data.

7 Conclusion

In this work, we proposed a novel neural network method to incorporate unlabeled and partially-labeled data for cross-domain Chinese word segmentation. To make use of unlabeled data, two character-level language models are integrated with the segmentation model through a gate mechanism. By modifying the loss function, the model could be trained using partially-labeled data. Experiments showed that incorporating these data led to significant improvements.

Acknowledgments

The authors wish to thank the anonymous reviewers for their helpful comments. This work was partially funded by China National Key R&D Program (No. 2017YFB1002104), National Natural Science Foundation of China (No. 61532011, 61751201, 61473092, and 61472088), and STCSM (No. 16JC1420401, 17JC1420200).

References

[Cai and Zhao, 2016] Deng Cai and Hai Zhao. Neural word segmentation learning for chinese. In *ACL*, 2016.

[Cai *et al.*, 2017] Deng Cai, Hai Zhao, Zhisong Zhang, Yuan Xin, Yongjian Wu, and Feiyue Huang. Fast and accurate neural word segmentation for chinese. *arXiv preprint arXiv:1704.07047*, 2017.

[Chen *et al.*, 2015] Xinchi Chen, Xipeng Qiu, Chenxi Zhu, Pengfei Liu, and Xuanjing Huang. Long short-term memory neural networks for chinese word segmentation. In *EMNLP*, 2015.

[Cho *et al.*, 2014] Kyunghyun Cho, Bart Van Merriënboer, Dzmitry Bahdanau, and Yoshua Bengio. On the properties of neural machine translation: Encoder-decoder approaches. *arXiv preprint arXiv:1409.1259*, 2014.

[Hochreiter and Schmidhuber, 1997] Sepp Hochreiter and Jürgen Schmidhuber. Long short-term memory. *Neural computation*, 1997.

[Huang *et al.*, 2007] Chu-Ren Huang, Petr Šimon, Shu-Kai Hsieh, and Laurent Prévot. Rethinking chinese word segmentation: tokenization, character classification, or word-break identification. In *ACL*, 2007.

[Huang *et al.*, 2015] Zhiheng Huang, Wei Xu, and Kai Yu. Bidirectional lstm-crf models for sequence tagging. *arXiv preprint arXiv:1508.01991*, 2015.

[Huang *et al.*, 2017] Shen Huang, Xu Sun, and Houfeng Wang. Addressing domain adaptation for chinese word segmentation with global recurrent structure. In *IJCNLP*, 2017.

[Jiang *et al.*, 2013] Wenbin Jiang, Meng Sun, Yajuan Lü, Yating Yang, and Qun Liu. Discriminative learning with natural annotations: Word segmentation as a case study. In *ACL*, 2013.

[Jin and Tanaka-Ishii, 2006] Zhihui Jin and Kumiko Tanaka-Ishii. Unsupervised segmentation of chinese text by use of branching entropy. In *COLING-ACL*, 2006.

[Kingma and Ba, 2014] Diederik Kingma and Jimmy Ba. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*, 2014.

[Li and Sun, 2009] Zhongguo Li and Maosong Sun. Punctuation as implicit annotations for chinese word segmentation. *Comput. Linguist.*, 2009.

[Liu and Zhang, 2012] Yang Liu and Yue Zhang. Unsupervised domain adaptation for joint segmentation and pos-tagging. In *COLING*, 2012.

[Liu *et al.*, 2014] Yijia Liu, Yue Zhang, Wanxiang Che, Ting Liu, and Fan Wu. Domain adaptation for crf-based chinese word segmentation using free annotations. In *EMNLP*, 2014.

[Liu *et al.*, 2016] Yijia Liu, Wanxiang Che, Jiang Guo, Bing Qin, and Ting Liu. Exploring segment representations for neural segmentation models. In *IJCAI*, 2016.

[Peters *et al.*, 2017] Matthew Peters, Waleed Ammar, Chandra Bhagavatula, and Russell Power. Semi-supervised sequence tagging with bidirectional language models. In *ACL*, 2017.

[Rei, 2017] Marek Rei. Semi-supervised multitask learning for sequence labeling. In *ACL*, 2017.

[Shen *et al.*, 2013] Mo Shen, Daisuke Kawahara, and Sadao Kurohashi. Chinese word segmentation by mining maximized substrings. In *IJCNLP*, 2013.

[Sun and Xu, 2011] Weiwei Sun and Jia Xu. Enhancing chinese word segmentation using unlabeled data. In *EMNLP*, 2011.

[Triggs and Verbeek, 2008] Bill Triggs and Jakob J Verbeek. Scene segmentation with crfs learned from partially labeled images. In *NIPS*, 2008.

[Tsuboi *et al.*, 2008] Yuta Tsuboi, Hisashi Kashima, Hiroki Oda, Shinsuke Mori, and Yuji Matsumoto. Training conditional random fields using incomplete annotations. In *COLING*, 2008.

[Xue, 2003] Nianwen Xue. Chinese word segmentation as character tagging. *Computational Linguistics and Chinese Language Processing*, 2003.

[Yang and Vozila, 2014] Fan Yang and Paul Vozila. Semi-supervised chinese word segmentation using partial-label learning with conditional random fields. In *EMNLP*, 2014.

[Yang *et al.*, 2017] Jie Yang, Yue Zhang, and Fei Dong. Neural word segmentation with rich pretraining. *arXiv preprint arXiv:1704.08960*, 2017.

[Zhang *et al.*, 2013] Longkai Zhang, Li Li, Zhengyan He, Houfeng Wang, and Ni Sun. Improving chinese word segmentation on micro-blog using rich punctuations. In *ACL*, 2013.

[Zhang *et al.*, 2014] Meishan Zhang, Yue Zhang, Wanxiang Che, and Ting Liu. Type-supervised domain adaptation for joint segmentation and pos-tagging. In *EACL*, 2014.

[Zhang *et al.*, 2016] Meishan Zhang, Yue Zhang, and Guohong Fu. Transition-based neural word segmentation. In *ACL*, 2016.

[Zhao and Kit, 2008] Hai Zhao and Chunyu Kit. An empirical comparison of goodness measures for unsupervised chinese word segmentation with a unified framework. In *IJCNLP*, 2008.

[Zhou *et al.*, 2017] Hao Zhou, Zhenting Yu, Yue Zhang, Shujian Huang, XIN-YU DAI, and Jiajun Chen. Word-context character embeddings for chinese word segmentation. In *EMNLP*, 2017.