# Transfer Learning for Low-Resource Chinese Word Segmentation with a Novel Neural Network

**Jingjing Xu** and **Xu Sun**
MOE Key Laboratory of Computational Linguistics, Peking University
School of Electronics Engineering and Computer Science, Peking University
{jingjingxu, xusun}@pku.edu.cn

arXiv:1702.04488v4 [cs.CL] 17 May 2017

## Abstract

Recent works have been shown effective in using neural networks for Chinese word segmentation. However, these models rely on large-scale data and are less effective for low-resource datasets because of insufficient training data. Thus, we propose a transfer learning method to improve low-resource word segmentation by leveraging high-resource corpora. First, we train a teacher model on high-resource corpora and then use the learned knowledge to initialize a student model. Second, a weighted data similarity method is proposed to train the student model on low-resource data with the help of high-resource corpora. Finally, given that insufficient data puts forward higher requirements for feature extraction, we propose a novel neural network which improves feature learning. Experiment results show that our work significantly improves the performance on low-resource datasets: 2.3% and 1.5% F-score on PKU and CTB datasets. Furthermore, this paper achieves state-of-the-art results: 96.1%, and 96.2% F-score on PKU and CTB datasets[1]. Besides, we explore an asynchronous parallel method on neural word segmentation to speed up training. The parallel method accelerates training substantially and is almost five times faster than a serial mode.

## 1 Introduction

Chinese word segmentation (CWS) is an important step in Chinese natural language processing. The most widely used approaches (Xue and Shen, 2003; Peng et al., 2004) treat CWS as a sequence labelling problem in which each character is assigned with a tag. Formally, given an input sequence $\mathbf{x} = x_1x_2...x_n$, it produces a tag sequence $\mathbf{y} = y_1y_2...y_n$. Many exsiting techniques, such as conditional random fields, have been successfully applied to CWS (Lafferty et al., 2001; Tseng, 2005; Zhao et al., 2010; Sun and Xu, 2011; Sun et al., 2014). However, these approaches incorporate many handcrafted features. Therefore, the generalization ability is restricted.

In recent years, neural networks have become increasingly popular in CWS, which focused more on the ability of automated feature extraction. Collobert et al. (2011) developed a general neural architecture for sequence labelling tasks. Pei et al. (2014) used convolutioanl neural networks to capture local features within a fixed size window. Chen et al. (2015a) proposed gated recursive neural networks to model feature combinations. The gating mechanism was also used by Cai and Zhao (2016).

However, this success relies on massive labelled data and are less effective on low-resource datasets. The major problem is that a small amount of labelled data leads to inadequate training and negatively impacts the ability of generalization. However, there are enough corpora which consist of massive annotated texts. All can be used to improve the task. Thus, we propose a transfer learning method to address the problem by leveraging high-resource datasets.

First, we train a teacher model on high-resource datasets and then use the learned knowledge to initialize a student model. Previous neural network models usually use random initialization which relies on massive labelled data. It is hard for a randomly initialized model to achieve the expected results on low-resource datasets. Motivated by

[1] Our code is publicly available at https://github.com/jingjingxupku/Low-Resource-CWS-

| Datasets | Training Size | Testing Size | Models | F-score |
|---|---|---|---|---|
| MSR | 86919 | 3985 | Zhang and Clark (2007) | 97.2 |
| | | | Sun et al. (2012) | 97.4 |
| | | | Pei et al. (2014) | 97.2 |
| | | | Cai and Zhao (2016) | 96.5 |
| PKU | 19055 | 1945 | Zhang and Clark (2007) | 94.5 |
| | | | Sun et al. (2012) | 95.4 |
| | | | Pei et al. (2014) | 95.2 |
| | | | Cai and Zhao (2016) | 95.5 |

Table 1: Results of previous neural network models on MSR and PKU datasets. For a low-resource dataset: PKU, there is a clear result decline compared with a high-resource dataset: MSR.

that, we propose a teacher-student framework to initialize the student model. Second, the student model is trained on a low-resource dataset with the help of high-resource corpora. However, it is hard to directly make use of high-resource datasets to train the student model because different corpora have different data distributions. The shift of data distributions is a major problem. To address the problem, we propose a weighted data similarity method which computes a similarity of each high-resource sample with a low-resource dataset. Experiment results show that using our transfer learning method, we substantially improve results on low-resource datasets.

On the other hand, insufficient data puts forward higher requirements for feature extraction. A single feature trained on insufficient training data is weak. Our key idea is to combine several kinds of weak features to achieve the better performance. Motivated by that, we propose an unified global-local neural network (UGL) in this paper and apply our transfer learning method to it. Compared with previous networks, our network has an advantage of capturing abundant and useful features. Unlike our transfer learning method which is proposed from the training's point of view, the new neural network is proposed to improve the task in view of the model structure. Experiment results show that our model achieves large improvements on low-resource datasets.

With the increase of layers which are designed to improve the ability of feature extraction, the training speed is becoming a limit. To speed up training, we explore mini-batch asynchronous parallel (MAP) learning on neural segmentation in this paper. Existing asynchronous parallel learning methods are mainly for sparse models (Recht et al., 2011; Mcmahan and Streeter, 2014). For dense models, like neural networks, asynchronous parallel methods bring inevitable gradient noises. However, the theoretical analysis by Sun (2016) showed that the learning process with gradient errors can still be convergent on neural models. Motivated by that, we explore the MAP approach on neural segmentation in this paper. The parallel method accelerates training substantially and is almost five times faster than a serial mode.

The main contributions of the paper are as follows:

- A transfer learning method is proposed to improve low-resource word segmentation by leveraging high-resource corpora.
- The transfer learning method is realized through a novel neural network which improves feature learning.
- To speed up training, we explore mini-batch asynchronous parallel learning on neural word segmentation in this paper.

## 2 Transfer Learning by Leveraging High-Resource Datasets

Table 1 shows that previous neural word segmentation models are less effective on low-resource datasets since these models only focus on in-domain supervised learning. Furthermore, there are enough corpora which consist of massive annotated texts. For scenarios where we have insufficient labelled data, transfer learning is an effective way to improve the task. Motivated by that, we propose a transfer learning method to leverage high-resource corpora.

First, we propose a teacher-student framework to initialize a model with the learned knowledge. We train a teacher model on a dataset where there is a large amount of training data (e.g., MSR). The learned parameters are used to initialize a student model. Therefore, the student model is trained from the learned parameters, rather than randomly initialization.

Second, the student model is trained on low-resource datasets with the help of high-resource corpora. However, since different corpora have different data distributions, it is hard to directly make use of high-resource datasets to train the student model. Thus, to avoid the shift of data distributions, high-resource corpora are used to train the student model based on the weighted data similarity method. This method identifies the similarity of each high-resource sample with a low-resource dataset. We use different learning rates for different samples. A learning rate is adjusted by the weighted data similarity automatically. The weighted data similarity $w_i^t$ is updated as follows.

First, calculate the update rate $a^t$:

$$e^t = (1 - \frac{2 * p^t * r^t}{p^t + r^t}) \tag{1}$$

$$a^t = \frac{1}{2} log \frac{1 - e^t}{e^t} \tag{2}$$

where $p^t$ and $r^t$ are precision and recall of the student model on high-resource data. The update rate $a^t$ is determined by the error rate $e^t$. The error rate is a simple and effective way to evaluate the data similarity.

Next, update the data similarity after $t$ iterations:

$$S^{t+1} = (w_1^{t+1}, \ldots, w_i^{t+1}, \ldots, w_N^{t+1}) \tag{3}$$

$$w_i^{t+1} = \frac{w_i^t}{Z^t * m} \sum_{j=1}^{m} exp(a^t I(y_{i,j} = p_{i,j})) \tag{4}$$

where $m$ is the length of sample $i$, $I()$ is the indicator function which evaluates if the prediction $p_{i,j}$ is equal with the gold label $y_{i,j}$ and $Z^t$ is the regularization factor which is computed as:

$$Z^t = \sum_i \frac{w_i^t}{m} \sum_{j=1}^{m} exp(a^t I(y_{i,j} = p_{i,j})) \tag{5}$$

Finally, the weighted data similarity is used to compute the learning rate $\alpha_i^t$:

$$\alpha_i^t = \alpha^t * w_i^t \tag{6}$$

where $\alpha^t$ is the fixed learning rate for a low-resource dataset, $w_i^t$ indicates the similarity between sentence $i$ and a low-resource corpus, which is from 0 to 1.

## 3 Unified Global-Local Neural Networks for Feature Extraction

Insufficient data puts forward higher requirements for feature extraction. Our key idea is to combine several kinds of weak features to achieve the better performance. Unlike previous networks which focus on a single kind of feature: either complicated local features or global dependencies, our network has an advantage of combing complicated local features with long dependencies together. Both of them are necessary for CWS and should not be neglected. Our network is built on a simple encoder-decoder structure. A encoder is designed to model local combinations and a decoder is used to capture long distance dependencies.

Figure 1 illustrates the model architecture. First, words are represented by embeddings stored in a lookup table $D^{|v|*d}$ where $v$ is the number of words in the vocabulary and $d$ is the embedding size. The lookup table is pre-trained on gigaword corpus where unknown words are mapped to a special symbol. The inputs to our model are $x_1, x_2, ..., x_n$ which are represented by $D = D_{x_1}, D_{x_{2,...,D_{x_n}}}$.

We first extract a window context $H^0 \in R^{n,k,d}$ from an input sequence which is padded with special symbols according to the window size:

$$H_{i,j}^0 = D[i + j] \tag{7}$$

where $n$ is the sentence length, $k$ is the window size and $d$ is the embedding length. $H^0$ will be input to the encoder to produce complicated local feature representations.

**Encoder.** The encoder is composed by filter recursive networks. Unlike GRNN (Chen et al., 2015a) which has a limit that inputs must be two vectors, filter recursive networks are proposed to break this limit by introducing filter mechanism which controls the input size. Motivated by Chen et al. (2015a) and Cai and Zhao (2016), the gating mechanism has been shown effective to model feature combinations. Thus, we adjust the gate function to fit our model. According to the filter size, we first choose every patch and input it to gate function to get next layer $H^1 \in R^{n,k-f_1+1,d}$ where $f_1$ is the filter size of $1^{th}$ hidden layer.

In a gate cell of filter recursive networks, output $H^1$ of the $i^{th}, j^{th}$ hidden node is computed as:

$$H_{i,j}^1 = z_h \odot h' + \sum_{d_i=0}^{f_1-1} (z_{d_i} \odot H_{i,j+d_i}^0) \tag{8}$$

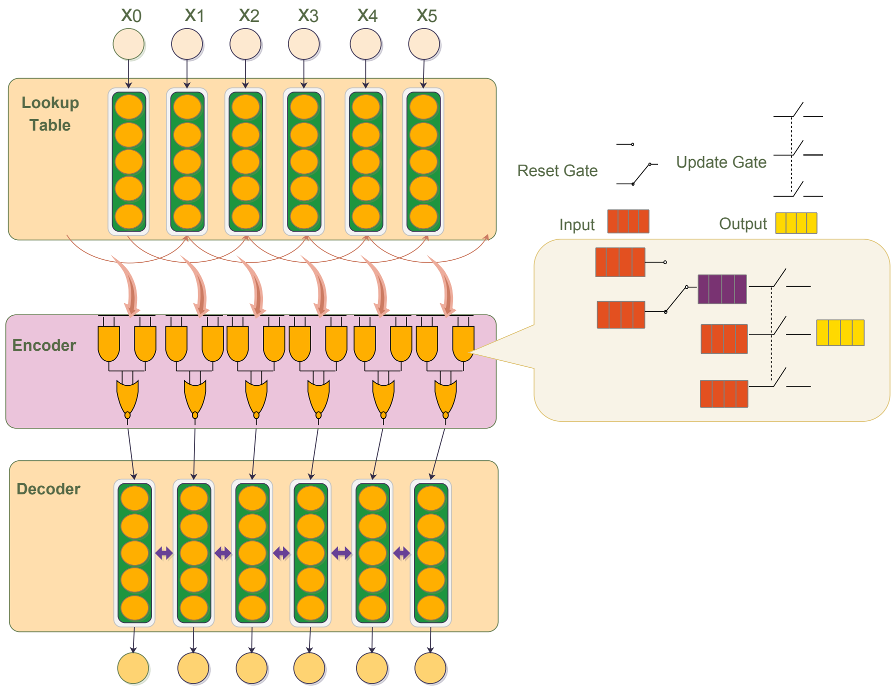



Figure 1: The architecture of unified global-local neural networks for CWS. The window size is 3 and the filter size is 2 in the example.

where $z_h$ and $z_{d_i}$ are update gates for new activation $h^{'}$ and inputs, while $\odot$ means element-wise multiplication. To simplify the cell, $z_h$, $z_{d_i}$ are computed as:

$$\begin{bmatrix} z_h \\ z_0 \\ \dots \\ z_{d_i} \\ \dots \\ z_{f_1-1} \end{bmatrix} = sigmoid(U \begin{bmatrix} h^{'} \\ H^0_{i,j} \\ \dots \\ H^0_{i,j+d_i} \\ \dots \\ H^0_{i,j+f_1-1} \end{bmatrix}) \quad (9)$$

where $U \in R_{(f_1+1)d*(f_1+1)d}$ and the new activation $h^{'}$ is computed as:

$$h^{'} = tanh(W \begin{bmatrix} r_0 \odot H^0_{i,j} \\ \dots \\ r_{d_i} \odot H^0_{i,j+d_i} \\ \dots \\ r_{f_1-1} \odot H^0_{i,j+f_1-1} \end{bmatrix}) \quad (10)$$

where $W \in R_{d*f_1d}$ and $r_0, ..., r_{f_1-1}$ are reset gates for inputs, which can be formalized as:

$$\begin{bmatrix} r_0 \\ \dots \\ r_{d_i} \\ \dots \\ r_{f_1-1} \end{bmatrix} = sigmoid(G \begin{bmatrix} H^0_{i,j} \\ \dots \\ H^0_{i,j+d_i} \\ \dots \\ H^0_{i,j+f_1-1} \end{bmatrix}) \quad (11)$$

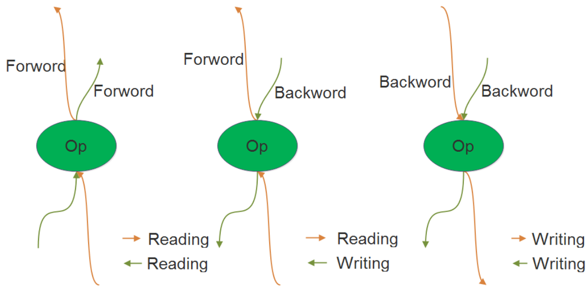



Figure 2: Three kinds of gradient problems: read-read, read-write and write-write conflicts.

These operations will repeat until we get $H^l \in R^{n,1,d}$ which is reduced dimension to $H^l \in R^{n,d}$.

**Decoder.** The decoder is composed by bi-directional long shor-term memory network (Bi-LSTM) (Hochreiter and Schmidhuber 1997). The local features encoded by filter recursive neural networks are refined into global dependencies and then decoded to tag sequences in this stage.

## 4 Mini-Batch Asynchronous Parallel Learning

With the development of multicore computers, there is a growing interest in parallel techniques.

Researchers have proposed several schemes (Zhao and Huang, 2013; Zinkevich et al., 2010), but most of them require locking so the speedup is limited. Asynchronous parallel learning methods without locking can maximize the speedup ratio. However, existing asynchronous parallel learning methods are mainly for sparse models. For dense models, like neural networks, asynchronous parallel learning brings gradient noises which are very common and inevitable (Figure 2). Read-read conflicts break the sequentiality of training procedure, read-write and write-write conflicts lead to incorrect gradients. Nevertheless, Sun (2016) proved that the learning process with gradient errors can still be convergent. Motivated by that, we train our model in the asynchronous parallel way.

We find that Adam (Kingma and Ba, 2014) is a practical method to train large neural networks. Then, we run the asynchronous parallel method on Adam training algorithm.

First, it recursively calculates $m_t$ and $v_t$, based on the gradient $g_t$. $\beta_1$ and $\beta_2$ are used to control the ratio of previous states.

$$m_t = \beta_1 \cdot m_{t-1} + (1 - \beta_1) \cdot g_t \tag{12}$$

$$v_t = \beta_2 \cdot v_{t-1} + (1 - \beta_2) \cdot g_t^2 \tag{13}$$

Second, $\Delta W_t$ is calculated based on $v_t$ and $m_t$. $\epsilon$ and $\mu$ are both smooth parameters.

$$M(w, t) = v_t - m_t^2 \tag{14}$$

$$\Delta w_t = \frac{\epsilon g_{t,i}}{\sqrt{M(w,t)} + \mu} \tag{15}$$

Finally, weight matrix is updated based on $\Delta w_t$. The parameter $\Theta_{t+1}$ at time step $t+1$ is:

$$\Theta_{t+1} = \Theta_t - \Delta w_t \tag{16}$$

Following experimental results on development datasets, hyper parameters of optimization method are set as follows: $\beta_1 = \beta_2 = 0.95$, $\epsilon = 1 \times 10^{-4}$.

The training algorithm is realized without any locking (see Table 2). For each mini-batch, we uniformly distribute it into different processors. Processors compute the increment of gradient $\Delta w_t$ in parallel, where $w_t$ is stored in a shared memory and each processor can read and update it.

We find that each processor always waits the end of the slowest processor because the length of sentence varies. Motivated by structure regularization (Sun, 2014), we split each sentence into several fixed-length mini-sentences. The task of each processor is almost the same, so the waiting time would be reduced greatly.

| **Mini-Batch Asynchronous Parallel Algorithm.** |
|---|
| Input: dataset $D$, mini-batch size $m$<br>and the number of processor $p$.<br>Output: weight matrix $\Theta_k$ on $k_{th}$ layer. |
| Split D into $\lceil n/m \rceil$ mini-batches<br>$S = \{D_1, D_2, D_3, ..., D_i, ..., D_{\lceil n/m \rceil}\}$<br>for $D_i$ in S:<br>  Split $D_i$ into $p$ processors<br>  Initializing $\Delta W$<br>  for $j$ in $[1, 2, ..., p]$ **in parallel**<br>    for sample $x$ in $D_{i,j}$:<br>      update $\Delta W + = \Delta w_x$<br>      **without locking**<br>    end for<br>  end for<br>  $\Theta_k$-= $\Delta W$<br>end for |

Table 2: The mini-batch asynchronous parallel algorithm.

| Datasets | Training Size | Testing Size |
|---|---|---|
| MSR | 86919 | 3985 |
| PKU | 19055 | 1945 |
| CTB | 68919 | 2193 |

Table 3: Details of different datasets. The size of training/testing datasets are given in number of sentences.

## 5 Experiments

The proposed model is evaluated on three datasets: MSR, PKU and CTB. Table 3 shows the details of these datasets. We treat MSR as a high-resource dataset, PKU and CTB as low-resurce datasets. MSR and PKU are provided by the second International Chinese Word Segmentation Bakeoff (Emerson, 2005). CTB is from Chinese TreeBank 8.0[2] and split to training and testing sets in this paper. We randomly split 10% of the training sets to development sets which are used to choose the suitable hyper-parameters.

All idioms, numbers and continuous English

[2]https://catalog.ldc.upenn.edu/LDC2013T21

| Models | PKU | | | CTB | | |
|---|---|---|---|---|---|---|
| | P | R | F | P | R | F |
| Bi-LSTM | 94.1 | 92.6 | 93.3 | 94.2 | 94.5 | 94.3 |
| GRNN | 94.5 | 93.6 | 94.0 | 94.8 | 94.9 | 94.8 |
| **UGL** | 95.2 | 94.1 | **94.6** | 95.4 | 95.2 | **95.3** |

Table 4: Comparisons between UGL and baselines on low-resource datasets: PKU and CTB.

| Models | PKU | CTB |
|---|---|---|
| Bi-LSTM | 93.3 | 94.3 |
| UGL | 94.6 | 95.3 |
| +Transfer Learning | **95.6** | **95.8** |
| **Improvement** | **2.3** | **1.5** |
| **Error Rate Reduction** | **34.3** | **26.3** |

Table 5: Improvements of our proposal on low-resource datasets: PKU and CTB.

characters are replaced to special flags. The improvements achieved by an idiom dictionary are very limited, less than 0.1% F-score on all datasets. The character embeddings are pretrained on Chinese gigaword[3] by word2vec[4].

All results are evaluated by $F_1$-score which is computed by the standard Bakeoff scoring program. Besides, we also consider an error rate reduction as one of the evaluation criterion. The error reduction evaluates what percentage of errors are corrected. The error rate reduction $R$ between model $a$ and model $b$ is calculated as follows:

$$R = \frac{e_a - e_b}{e_a} \tag{17}$$

where $e_a$ and $e_b$ are error rates of model $a$ and model $b$. $e_a$ and $e_b$ are computed as:

$$e_a = 1 - f_a \tag{18}$$

$$e_b = 1 - f_b \tag{19}$$

where $f_a$ and $f_b$ are F-score values of model $a$ and model $b$.

### 5.1 Setup

Hyper-parameters are set according to the performance on development sets. We evaluate the mini-batch size $m$ in a serial mode and choose $m = 16$. Similarly, the window size $w$ is set as 5, the fixed learning rate $\alpha$ is set as 0.01, the dimension of character embeddings and hidden layers $d$ is set as 100. $d = 100$ is a good balance between model speed and performance.

Inspired by Pei et al. (2014), bigram features are applied to our model as well. Specifically, each bigram embedding is represented as a single vector. Bigram embeddings are initialized randomly. We ignore lots of bigram features which only appear once or twice since these bigram features not only are useless, but also make a bigram lookup table huge.

All experiments are performed on a commodity 64-bit Dell Precision T5810 workstation with one 3.0GHz 16-core CPU and 64GB RAM. The C# multiprocessing module is used in this paper.

### 5.2 Results and Discussions

Table 5 shows the improvements of our proposal. The proposal is compared with Bi-LSTM which is a competitive and widely used model for neural word segmentation. Experiment result show that our proposal achieves substantial improvements on low-resource datasets: 2.3% and 1.5% F-score on PKU and CTB datasets. Besides, the error rate is decreased by 34.3% and 26.3%.

**Transfer Learning.** The improvements of transfer learning are shown in Table 5. We choose MSR as a high-resource dataset. Results on PKU and CTB datasets all show improvements: 1.0% F-score on PKU dataset and 0.5% on CTB dataset. A high-resource dataset not only decreases the number of out-of-vocabulary words, but also improves results of in-vocabulary words. The size of PKU dataset is far less than that of CTB dataset and we achieve the better improvements on PKU dataset. It shows that our transfer learning method is more efficient on datasets with fewer resource.

**Unified Global-Local Neural Networks.** We reconstruct some of state-of-the-art neural models in this paper: Bi-LSTM and GRNN. Bi-LSTM is used for capturing global and simple local features. GRNN is used for capturing complicated local features. Table 4 shows that our model out-

[3]https://catalog.ldc.upenn.edu/LDC2003T09
[4]https://code.google.com/archive/p/word2vec

| | Models | PKU | | | CTB | | |
|---|---|---|---|---|---|---|---|
| | | P | R | F | P | R | F |
| Unigram | Zheng et al. (2013) | 92.8 | 92.0 | 92.4 | * | * | * |
| | Pei et al. (2014) | 94.4 | 93.6 | 94.0 | * | * | * |
| | Cai and Zhao (2016) | 95.8 | 95.2 | 95.5 | * | * | * |
| | **Our Work** | 96.0 | 95.1 | **95.6** | 95.9 | 95.8 | **95.8** |
| Bigram | Pei et al. (2014) | * | * | 95.2 | * | * | * |
| | Ma and Hinrichs (2015) | * | * | 95.1 | * | * | * |
| | Zhang et al. (2016) | * | * | 95.7 | * | * | * |
| | **Our Work** | 96.3 | 95.9 | **96.1** | 96.2 | 96.1 | **96.2** |

Table 6: Comparisons with state-of-the-art neural networks on lower-resource datasets: PKU and CTB.

| Models | PKU | | | CTB | | |
|---|---|---|---|---|---|---|
| | P | R | F | P | R | F |
| Tseng et al. (2005) | * | * | 95.0 | * | * | * |
| Zhang et al. (2006) | * | * | 95.1 | * | * | * |
| Zhang and Clark (2007) | * | * | 94.5 | * | * | * |
| Sun et al. (2012) | * | * | 95.4 | * | * | * |
| **Our Work** | 96.3 | 95.9 | **96.1** | 96.2 | 96.1 | **96.2** |

Table 7: Comparisons with previous traditional models on lower-resource datasets: PKU and CTB.

| Models | P | R | F |
|---|---|---|---|
| Bi-LSTM | 95.1 | 94.7 | 94.9 |
| GRNN | 95.9 | 95.9 | 95.9 |
| **UGL** | 96.9 | 96.7 | **96.8** |

Table 8: Comparisons between our model and baselines on a high-resource dataset: MSR.

performs baselines on low-resource datasets: PKU and CTB. It proves that combining several weak features is an effective way to improve the performance on low-resource datasets.

Besides, we compare our model with baselines on a high-resource dataset: MSR. Table 8 shows that our model achieves the best results. It can prove that our model is also applied to high-resource datasets.

**Comparisons with State-of-the-art Models.** Table 6 shows comparisons between our work and latest neural models on low-resource datasets: PKU and CTB. Experiment results show that our work largely outperforms state-of-the-art models which are very competitive. Given that a dictionary used in Chen et al. (2015a) is not publicly released, our work is not comparable with it.

We also compare our work with traditional models on low-resource datasets: PKU and CTB, several of which take advantage of a variety of feature templates and dictionaries. As shown in Table 7, our work achieves state-of-the-art results.

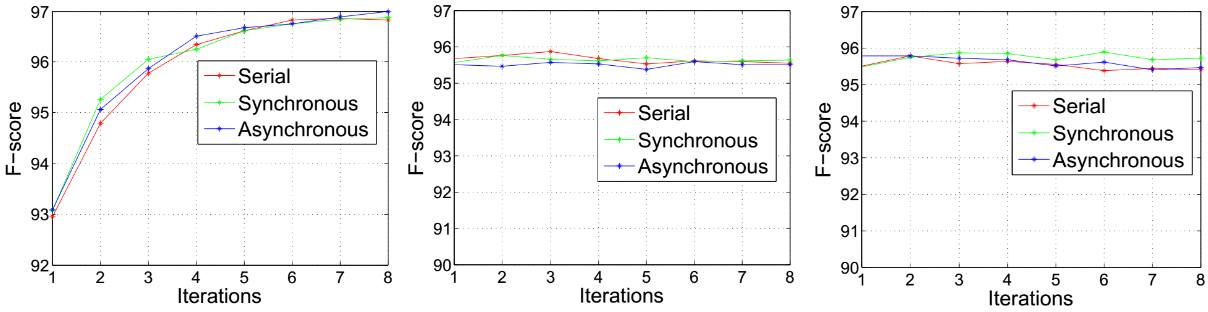
Although our model only uses simple bigram features, it outperforms the previous state-of-the-art methods which use more complex features.

**Mini-Batch Asynchronous Parallel Learning.** We run the proposed model in asynchronous, synchronous and serial modes to analyze the parallel efficiency. The number of threads used in asynchronous and synchronous modes is 15. The comparisons are shown in Figure 3 and 4. It can be clearly seen that the asynchronous algorithm achieves the best speedup ratio without decreasing F-score compared with synchronous and serial algorithms. The asynchronous parallel algorithm is almost 5x faster than the serial algorithm.

## 6 Related Work

Next, we briefly review neural word segmentation, transfer learning in CWS and asynchronous parallel learning.

**Neural Word Segmentation.** Zheng et al. (2013) used a two-layer network and adapted a general neural network architecture in Collobert et al. (2011). Pei et al. (2014) used a tensor framework to capture feature combination. Chen et al. (2015a) proposed gated recursive neural networks to model feature combinations of context characters. Chen et al. (2015b) used LSTM to model long distance dependencies in a sentence.




(a) MSR (b) PKU (c) CTB

Figure 3: Comparisons of F-score performance among serial, synchronous and asynchronous algorithms on three datasets.

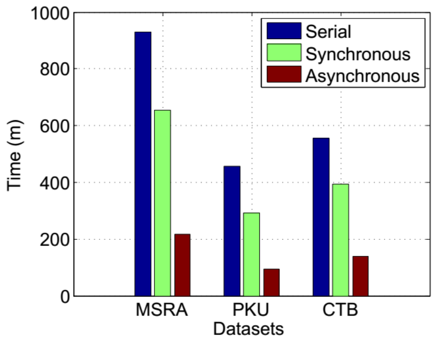



Figure 4: Comparisons of training time among serial, synchronous and asynchronous algorithms on three datasets.

Zhang et al. (2016) applied the transition-based neural framework to Chinese segmentation. Cai and Zhao (2016) proposed a novel neural framework which thoroughly eliminated context windows and utilized complete segmentation history.

**Transfer Learning in CWS.** Sun and Xu (2011) introduced many statistical features from unlabeled in-domain data to enhance supervised method in CWS. Zhang et al.(2014) used type-supervised domain adaptation for joint Chinese word segmentation and POS-tagging. Liu et al. (2014) adopted freely available data to help improve the performance on CWS. However, these transfer learning methods involved in too many handcrafted features. In our work, deep neural networks are used to reduce handcrafted efforts.

**Asynchronous Parallel Learning.** Recently, a viriety of asynchronous parallel learning methods have been developed (Recht et al., 2011; Mcmahan and Streeter, 2014). Those asynchronous methods have shown to be more effective than synchronous parallel learning. However, existing asynchronous parallel learning methods are mainly for sparse parameter models to avoid the problem of gradient error. For dense parameter models like neural networks, asynchronous parallel methods bring gradient errors. The theoretical analysis work of Sun (2016) showed that the learning process with gradient errors can still be convergent on neural models.

## 7 Conclusions

The major problem of low-resource word segmentation is insufficient training data. Thus, we propose a transfer learning method to improve the task by leveraging high-resource datasets. First, it is hard for a randomly initialized model to achieve the expected results on low-resource datasets. Therefore, we propose a teacher-student framework to initialize a student model. Second, the student model is trained on a low-resource dataset with the help of high-resource corpora. A weighted data similarity is proposed to avoid the shift of data distributions. Our transfer learning method is realised through a novel neural network which improves feature learning. Experiment results show that our work largely improves the performance on low-resource datasets compared with state-of-the-art models. Finally, our parallel training method brings substantial speedup and is almost 5x faster than a serial mode.

## References

Deng Cai and Hai Zhao. 2016. Neural word segmentation learning for chinese. In *Meeting of the Association for Computational Linguistics*.

Xinchi Chen, Xipeng Qiu, Chenxi Zhu, and Xuanjing Huang. 2015a. Gated recursive neural network for chinese word segmentation. In *ACL (1)*. The Association for Computer Linguistics, pages 1744–1753.

Xinchi Chen, Xipeng Qiu, Chenxi Zhu, Pengfei Liu, and Xuanjing Huang. 2015b. Long short-term memory neural networks for chinese word segmentation. In *EMNLP*. The Association for Computational Linguistics, pages 1197–1206.

Ronan Collobert, Jason Weston, Léon Bottou, Michael Karlen, Koray Kavukcuoglu, and Pavel Kuksa. 2011. Natural language processing (almost) from scratch. *J. Mach. Learn. Res.* 12:2493–2537.

Thomas Emerson. 2005. The second international chinese word segmentation bakeoff. In *Proceedings of the Fourth SIGHAN Workshop on Chinese Language Processing*. pages 123–133.

Diederik Kingma and Jimmy Ba. 2014. Adam: A method for stochastic optimization. *Computer Science* .

John D. Lafferty, Andrew McCallum, and Fernando C. N. Pereira. 2001. Conditional random fields: Probabilistic models for segmenting and labeling sequence data. In *Proceedings of the Eighteenth International Conference on Machine Learning*. Number 8 in ICML '01, pages 282–289.

Yijia Liu, Yue Zhang, Wanxiang Che, Ting Liu, and Fan Wu. 2014. Domain adaptation for crf-based chinese word segmentation using free annotations. In Alessandro Moschitti, Bo Pang, and Walter Daelemans, editors, *EMNLP*. ACL, pages 864–874.

Jianqiang Ma and Erhard W Hinrichs. 2015. Accurate linear-time chinese word segmentation via embedding matching. In *ACL (1)*. pages 1733–1743.

H. B. Mcmahan and M. Streeter. 2014. Delay-tolerant algorithms for asynchronous distributed online learning. *Advances in Neural Information Processing Systems* 4:2915–2923.

Wenzhe Pei, Tao Ge, and Baobao Chang. 2014. Max-margin tensor neural network for chinese word segmentation. In *Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*. Association for Computational Linguistics, Baltimore, Maryland, pages 293–303.

Fuchun Peng, Fangfang Feng, and Andrew McCallum. 2004. Chinese segmentation and new word detection using conditional random fields. In *Proceedings of the 20th International Conference on Computational Linguistics*. Association for Computational Linguistics, Stroudsburg, PA, USA, COLING '04.

Benjamin Recht, Christopher Ré, Stephen J. Wright, and Feng Niu. 2011. Hogwild: A lock-free approach to parallelizing stochastic gradient descent. In *NIPS*. pages 693–701.

Weiwei Sun and Jia Xu. 2011. Enhancing chinese word segmentation using unlabeled data. In *Conference on Empirical Methods in Natural Language Processing, EMNLP 2011, 27-31 July 2011, John Mcintyre Conference Centre, Edinburgh, Uk, A Meeting of Sigdat, A Special Interest Group of the ACL*. pages 970–979.

Xu Sun. 2014. Structure regularization for structured prediction. In *Advances in Neural Information Processing Systems 27*. pages 2402–2410.

Xu Sun. 2016. Asynchronous parallel learning for neural networks and structured models with dense features. In *COLING*.

Xu Sun, Wenjie Li, Houfeng Wang, and Qin Lu. 2014. Feature-frequency-adaptive on-line training for fast and accurate natural language processing. *Computational Linguistics* 40(3):563–586.

Xu Sun, Houfeng Wang, and Wenjie Li. 2012. 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)*. Association for Computational Linguistics, Jeju Island, Korea, pages 253–262.

Huihsin Tseng. 2005. A conditional random field word segmenter. In *In Fourth SIGHAN Workshop on Chinese Language Processing*.

N. Xue and L. Shen. 2003. Chinese Word Segmentation as LMR Tagging. In *Proceedings of the 2nd SIGHAN Workshop on Chinese Language Processing*.

Meishan Zhang, Yue Zhang, Wanxiang Che, and Ting Liu. 2014. Type-supervised domain adaptation for joint segmentation and pos-tagging. In *EACL*. pages 588–597.

Meishan Zhang, Yue Zhang, and Guohong Fu. 2016. Transition-based neural word segmentation. In *Meeting of the Association for Computational Linguistics*. pages 421–431.

Ruiqiang Zhang, Genichiro Kikui, and Eiichiro Sumita. 2006. Subword-based tagging by conditional random fields for chinese word segmentation. In *Proceedings of the Human Language Technology Conference of the NAACL, Companion Volume: Short Papers*. Association for Computational Linguistics, Stroudsburg, PA, USA, NAACL-Short '06, pages 193–196.

Yue Zhang and Stephen Clark. 2007. Chinese segmentation with a word-based perceptron algorithm. In *Proceedings of the 45th Annual Meeting of the Association of Computational Linguistics*. Association for Computational Linguistics, Prague, Czech Republic, pages 840–847.

Hai Zhao, Changning Huang, Mu Li, and Bao-Liang Lu. 2010. A unified character-based tagging framework for chinese word segmentation. *ACM Trans. Asian Lang. Inf. Process.* 9(2).

Kai Zhao and Liang Huang. 2013. Minibatch and parallelization for online large margin structured learning. In *HLT-NAACL*. The Association for Computational Linguistics, pages 370–379.

Xiaoqing Zheng, Hanyang Chen, and Tianyu Xu. 2013. Deep learning for chinese word segmentation and pos tagging. In *EMNLP*. ACL, pages 647–657.

Martin Zinkevich, Markus Weimer, Lihong Li, and Alex J. Smola. 2010. Parallelized stochastic gradient descent. In J.D. Lafferty, C.K.I. Williams, J. Shawe-Taylor, R.S. Zemel, and A. Culotta, editors, *Advances in Neural Information Processing Systems 23*, Curran Associates, Inc., pages 2595–2603.