Proceedings of the Twenty-Sixth International Joint Conference on Artificial Intelligence (IJCAI-17)

A Feature-Enriched Neural Model for Joint Chinese Word Segmentation and Part-of-Speech Tagging

Xinchi Chen, Xipeng Qiu*, **Xuanjing Huang**
Shanghai Key Laboratory of Intelligent Information Processing, Fudan University
School of Computer Science, Fudan University
825 Zhangheng Road, Shanghai, China
{xinchichen13,xpqiu,xjhuang}@fudan.edu.cn

Abstract

Recently, neural network models for natural language processing tasks have been increasingly focused on for their ability of alleviating the burden of manual feature engineering. However, the previous neural models cannot extract the complicated feature compositions as the traditional methods with discrete features. In this work, we propose a feature-enriched neural model for joint Chinese word segmentation and part-of-speech tagging task. Specifically, to simulate the feature templates of traditional discrete feature based models, we use different filters to model the complex compositional features with convolutional and pooling layer, and then utilize long distance dependency information with recurrent layer. Experimental results on five different datasets show the effectiveness of our proposed model.

1 Introduction

Chinese word segmentation and part-of-speech (POS) tagging are two core and fundamental tasks in Chinese natural language processing (NLP). The state-of-the-art approaches are based on joint segmentation and tagging (S&T) model, which can be regarded as character based sequence labeling task. The joint model can alleviate the error propagation problem of pipeline models.

Previously, the traditional hand-crafted feature based models have achieved great success on joint S&T task [Jiang *et al.*, 2008; Kruengkrai *et al.*, 2009; Qian *et al.*, 2010; Zhang and Clark, 2008; 2010]. Despite of their success, their performances are easily affected by following two limitations.

The first is **model complexity**. Since the decoding space of joint S&T task is relatively large, the traditional models often rely on millions of discrete features. Therefore, the efficiency of joint S&T models is rather low. Moreover, these models suffer from data sparsity. Recently, some neural models [Huang *et al.*, 2015; Chen *et al.*, 2015a; Ma and Hovy, 2016] are proposed to reduce the efforts of feature engineering and the model complexity. However, these neural models just concatenate the embeddings of the context characters, and feed them into neural network. Since the concatenation operation is relatively simple, it is difficult to model the complicated features as the traditional discrete feature based models. Although the complicated interactions of inputs can be modeled by the deep neural network, the previous neural models show that the deep model cannot outperform the one with a single non-linear model.

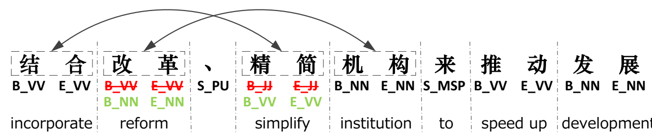

Figure 1: An example. CRF makes mistakes on words "改革 (reform)" and "精简 (simplify)". The red tags with strikethrough lines indicate the wrong predictions.

The second is **long term dependency**. Unlike pure POS tagging task which can utilize contextual features on word level, joint S&T task usually extracts the contextual features on character level. Thus, the joint model need longer dependency on character level. As the example shown in Figure 1, conditional random field (CRF) model makes mistakes on words "改革 (reform)" and " 精简 (simplify)" since it is hard for CRF to disambiguate the POS tags without using long distance information. However, restricted by model complexity and data sparsity, a larger window size (greater than 5) will instead hurt the performance. Therefore, how to exploit the long distance information without increasing the model complexity is crucial to joint S&T task.

In order to address these two problems, we propose a feature-enriched neural model for joint S&T task, which consists of several key components: (1) a convolutional layer to simulate compositional features as complex hand-crafting features; (2) a pooling layer to select the most valuable features; (3) a bi-directional long short-term memory (BLSTM) layer on the top to carry long distance dependency information. In addition, we introduce a highway layer [Srivastava *et al.*, 2015] to increase the depth of architecture and obtain more sophisticated feature representation without sufftering from the problem of gradient vanishing, leading to fast convergence.

Our contributions could be summaries as follows:

1. We propose a customized neural architecture for joint

* Corresponding author.

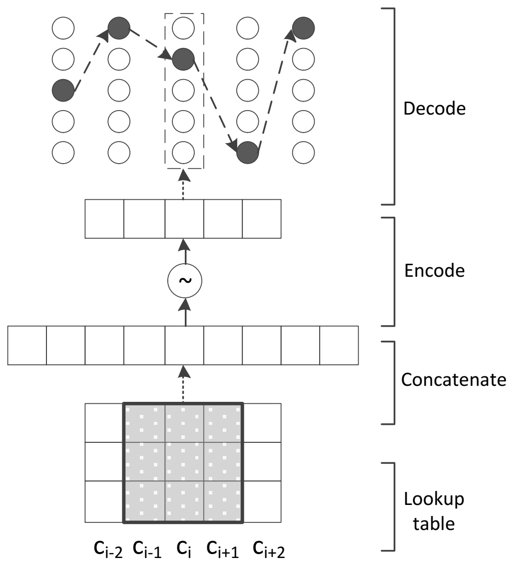

Figure 2: General neural network based architecture for joint S&T. The solid arrow denotes that there is a weight matrix on the link, while the dashed one denotes none.

S&T task, in which each component is designed according to its specific requirements, instead of a general deep neural model.

2. Our model can alleviate two crucial problems: model complexity and long term dependency in joint S&T task.
3. We evaluate our model on five different datasets. Experimental results show that our model achieves comparable performance to the previous sophisticated feature based models, and outperforms the previous neural models.

2 Neural Models for Joint S&T

The joint S&T task is usually regraded as a character based sequence labeling problem.

In this paper, we employ the {B M E S} tag set $\mathcal{T}_{SEG}$ (indicating the Begin, Middle, End of the word, and a Single character word respectively) for word segmentation and the tag set $\mathcal{T}_{POS}$ (varies from dataset to dataset) for POS tagging.

The tag set $\mathcal{T}$ of our joint S&T task would be the cross-label set of $\mathcal{T}_{SEG}$ and $\mathcal{T}_{POS}$. As illustrated in Figure 1, we would have a tag B_VV for character " 结", where B $\in \mathcal{T}_{SEG}$ and VV $\in \mathcal{T}_{POS}$, indicating the first character of the VV word "结合".

Conventional neural network based model for sequence labeling task usually consists of three phases. Figure 2 gives the illustration.

2.1 Lookup Table Phase

In order to represent characters as distributed vectors, we usually apply a feed-forward neural layer on the top of the one-hot character representations. The parameter matrix of the neural layer is called character embedding matrix $\mathbf{E} \in \mathbf{R}^{|C| \times d}$, where C is the character set and d is the dimensionality of the character embeddings. For a given sentence $c_{1:n}$ of length n, the first step is to lookup embeddings of the characters in the current window slide $c_{i-\lfloor\frac{k-1}{2}\rfloor:i+\lceil\frac{k-1}{2}\rceil}$ for the current character c_i which is going to be tagged, where k is a hyper-parameter indicating the window size. By concatenating the embeddings, we get the representation $\mathbf{x}_i$ for the current character c_i.

2.2 Encoding Phase

Usually, we apply a linear transformation followed by a non-linear function to the current input $\mathbf{x}_i$:

$$\mathbf{h}_i = \mathbf{g}(\mathbf{W}_h{}^{\intercal} \times \mathbf{x}_i + \mathbf{b}_h), \tag{1}$$

where $\mathbf{W}_h \in \mathbf{R}^{kd \times h}$ and $\mathbf{b}_h \in \mathbf{R}^h$ is the trainable parameters, and h is the dimensionality of the hidden layer, $\mathbf{g}(\cdot)$ is a non-linear function which could be $\mathbf{sigmoid}(\cdot)$, $\mathbf{tanh}(\cdot)$, etc.

Then, we could get the score vector $\mathbf{p}_i \in \mathbf{R}^{|\mathcal{T}|}$ for each possible tags of current character c_i by applying a linear transformation layer to the hidden layer $\mathbf{h}_i$:

$$\mathbf{p}_i = \mathbf{W}_p{}^{\intercal} \times \mathbf{h}_i + \mathbf{b}_p, \tag{2}$$

where $\mathbf{W}_p \in \mathbf{R}^{h \times |\mathcal{T}|}$ and $\mathbf{b}_p \in \mathbf{R}^{|\mathcal{T}|}$ is the trainable parameters, and $\mathcal{T}$ is the joint tag set.

2.3 Decoding Phase

The decoding phase aims to select the best tag sequence $\hat{t}_{1:n}$, to maximize the reward function $\mathbf{r}(\cdot)$:

$$\mathbf{r}(t_{1:n}) = \sum_{i=2}^{n} \left(\mathbf{A}_{t_{i-1}t_i}\right) + \sum_{i=1}^{n} \left(\mathbf{p}_i[t_i]\right), \tag{3}$$

$$\hat{t}_{1:n} = \underset{t_{1:n} \in \mathbf{T}(c_{1:n})}{\arg\max} \; \mathbf{r}(t_{1:n}), \tag{4}$$

where $\mathbf{A} \in \mathbf{R}^{|\mathcal{T}| \times |\mathcal{T}|}$ is the transition parameter, indicating how possible a label will transfer to another. $\mathbf{T}(c_{1:n})$ indicates all possible tag sequences for sentence $c_{1:n}$.

Also, we employ the Viterbi algorithm [Forney Jr, 1973] to decode the best tag sequence in polynomial time complexity.

3 A Feature-Enriched Neural Model for Joint S&T

The simple neural model presented above achieves good results on the joint S&T task. However, the simple neural model, who concatenates the embeddings of contextual characters as features, is not as strong as models based on the handcrafted features. Thus, a simple shallow neural is insufficient to tackle with ambiguous cases which rely on more sophisticated feature combinations and long distance dependencies.

To deal with these issues, we propose a feature-enriched neural model for joint S&T task, which consists of three different types of neural layers, stacked one by one: (1) Convolutional layer; (2) Highway layer; (3) Recurrent layer. Figure 3 gives the illustration.

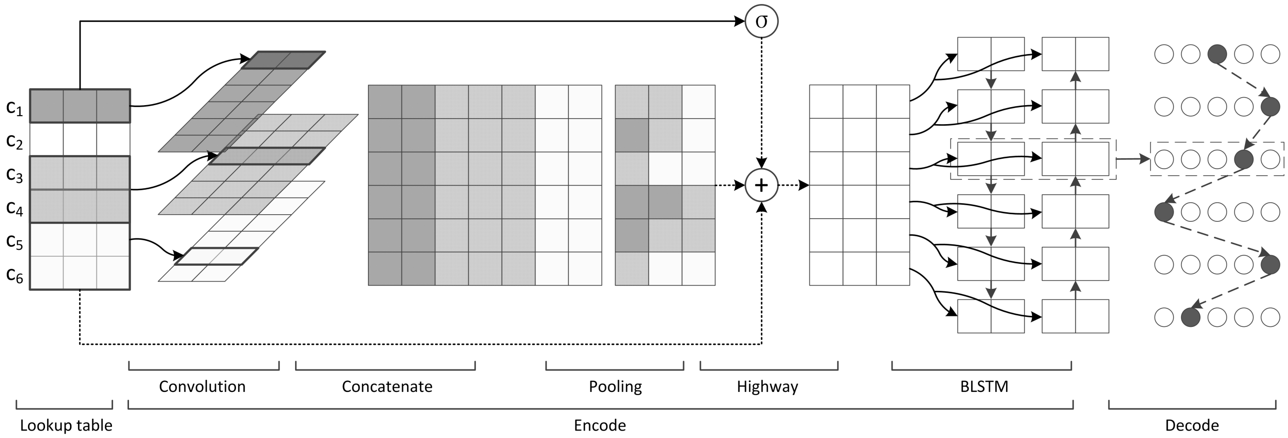

Figure 3: The proposed feature-enriched neural model for joint S&T task. The solid arrow denotes that there is a weight matrix on the link, while the dashed one denotes none.

3.1 Convolutional Layer

The simple neural model is just to concatenate the embeddings of characters in a local context, which cannot simulate the carefully designed features in traditional models.

To better model the complex compositional features as conventional feature based models, we use convolution layer to separately model different n-gram features for each character. Thus the feature of each character is the concatenation of corresponding columns of all different feature map sets. Then we apply a k-max pooling layer to select the most significant signals.

Concretely, we model uni-gram, bi-gram, ..., Q-gram features by generating feature map sets $\hat{\mathbf{z}}^1$, $\hat{\mathbf{z}}^2$, ..., $\hat{\mathbf{z}}^Q$ correspondingly. Formally, the q-gram feature map set $\hat{\mathbf{z}}^q$ is:

$$\hat{\mathbf{z}}_i^q = \tanh(\mathbf{W}_{cov}^{q}{}^{\mathsf{T}} \times \mathbf{x}_{i-\lfloor\frac{q-1}{2}\rfloor:i+\lceil\frac{q-1}{2}\rceil} + \mathbf{b}), i \in [1, n], \quad (5)$$

where $\mathbf{W}_{cov}^q \in \mathbf{R}^{qd \times l_q}$ is the convolutional filter for q-gram feature map set, and $\mathbf{x}_{i-\lfloor\frac{q-1}{2}\rfloor:i+\lceil\frac{q-1}{2}\rceil} \in \mathbf{R}^{qd}$ is the concatenation of embeddings of characters $c_{i-\lfloor\frac{q-1}{2}\rfloor:i+\lceil\frac{q-1}{2}\rceil}$. Here, l_q is the number of feature maps in q-gram feature map set and $\mathbf{b} \in \mathbf{R}^{l_q}$ is a bias parameter. For marginal cases, we use wide convolution strategy, which means we receive the sequence in the same length as input by padding zeros to the input.

Then, we would represent the original sentence by concatenation operation as $\mathbf{z} \in \mathbf{R}^{n \times \sum_{q=1}^{Q} l_q}$:

$$\mathbf{z}_i = \oplus_{q=1}^{Q} \hat{\mathbf{z}}_i, \quad (6)$$

where operator $\oplus$ is the concatenation operation.

After taking the k-max pooling operation, the representation of original sentence would be $\hat{\mathbf{X}} \in \mathbf{R}^{n \times d} = [\hat{\mathbf{x}}_1, \hat{\mathbf{x}}_2, \ldots, \hat{\mathbf{x}}_n]^{\mathsf{T}}$, where $\hat{\mathbf{x}}_i$ is:

$$\hat{\mathbf{x}}_i = k \max \mathbf{z}_i, k = d. \quad (7)$$

Hence, after convolutional layer, we would represent the given input sequence $\mathbf{X} \in \mathbf{R}^{n \times d} = [\mathbf{x}_1, \mathbf{x}_2, \ldots, \mathbf{x}_n]^{\mathsf{T}}$ as $\hat{\mathbf{X}} = Cov(\mathbf{X})$.

3.2 Highway Layer

Highway layer [Srivastava *et al.*, 2015] aims to keep gradient in very deep neural network. By introducing highway layer, we could simulate more complex compositional features by increasing the depth of our architecture. In addition, highway layer speeds up convergence speed and alleviates the problem of gradient vanishing.

As described above, we would represent the input sequence as $\hat{\mathbf{X}} = Cov(\mathbf{X})$ after the convolutional layer. By additionally adding the highway layer, the representation of the input sequence would be $\hat{\mathbf{X}}$ as:

$$\hat{\mathbf{X}} = Cov(\mathbf{X}) \odot T(\mathbf{X}) + \mathbf{X} \odot C(\mathbf{X}), \quad (8)$$

where operator $\odot$ indicates the element-wise multiplication operation. The $T(\cdot)$ is the transform gate and $C(\cdot)$ is the carry gate. We adopt a simple version, where we set $C(\cdot) = 1 - T(\cdot)$. Transform gate $T(\cdot)$ could be formalized as:

$$T(\mathbf{X}) = \sigma(\mathbf{W}_T{}^{\mathsf{T}} \times \mathbf{X} + \mathbf{b}_T), \quad (9)$$

where $\mathbf{W}_T \in \mathbf{R}^{d \times d}$ and $\mathbf{b}_T \in \mathbf{R}^d$ are trainable parameters. Here σ is the $sigmoid$ function.

3.3 Recurrent Layer

In joint S&T task, it usually relies on long distance dependency and sophisticated features to disambiguate lots of cases. Thus, a simple shallow neural model is insufficient to capture long distance information.

Inspired by recent works using long short-term memory (LSTM) [Hochreiter and Schmidhuber, 1997] neural networks, we utilize LSTM to capture the long-term and short-term dependencies. LSTM is an extension of the recurrent neural network (RNN) [Elman, 1990], which aims to avoid the problems of gradient vanishing and explosion, and is very suitable to carry the long dependency information.

By further adding LSTM layer on the top of $\hat{\mathbf{X}} \in \mathbf{R}^{n \times d} = [\hat{\mathbf{x}}_1, \hat{\mathbf{x}}_2, \ldots, \hat{\mathbf{x}}_n]$, we would represent sentence $c_{1:n}$ as $\mathbf{H} \in \mathbf{R}^{n \times h} = \mathbf{LSTM}(\hat{\mathbf{X}}) = [\mathbf{h}_1, \mathbf{h}_2, \ldots, \mathbf{h}_n]$. Specifically, LSTM layer introduces memory cell $\mathbf{c} \in \mathbf{R}^h$ which controlled by input gate $\mathbf{i} \in \mathbf{R}^h$, forget gate $\mathbf{f} \in \mathbf{R}^h$ and output gate $\mathbf{o} \in \mathbf{R}^h$. Thus, each output $\mathbf{h}_i \in \mathbf{R}^h$ would be calculated as:

$$\begin{bmatrix} \mathbf{i}_i \\ \mathbf{o}_i \\ \mathbf{f}_i \\ \hat{\mathbf{c}}_i \end{bmatrix} = \begin{bmatrix} \sigma \\ \sigma \\ \sigma \\ \phi \end{bmatrix} \left(\mathbf{W}_g{}^\intercal \begin{bmatrix} \hat{\mathbf{x}}_i \\ \mathbf{h}_{i-1} \end{bmatrix} + \mathbf{b}_g \right), \tag{10}$$

$$\mathbf{c}_i = \mathbf{c}_{i-1} \odot \mathbf{f}_i + \hat{\mathbf{c}}_i \odot \mathbf{i}_i, \tag{11}$$

$$\mathbf{h}_i = \mathbf{o}_i \odot \phi(\mathbf{c}_i), \tag{12}$$

where $\mathbf{W}_g \in \mathbf{R}^{4h \times (d+h)}$ and $\mathbf{b}_g \in \mathbf{R}^{4h}$ are trainable parameters. Here, the hyper-parameter h is dimensionality of $\mathbf{i}$, $\mathbf{o}$, $\mathbf{f}$, $\mathbf{c}$ and $\mathbf{h}$. $\sigma(\cdot)$ is $sigmoid$ function and $\phi(\cdot)$ is $tanh$ function.

BLSTM We also employ the bi-directional LSTM (BLSTM) neural network. Specifically, each hidden state of BLSTM is formalized as:

$$\mathbf{h}_i = \overrightarrow{\mathbf{h}}_i \oplus \overleftarrow{\mathbf{h}}_i, \tag{13}$$

where operator $\oplus$ indicates concatenation operation. Here, $\overrightarrow{\mathbf{h}}_i$ and $\overleftarrow{\mathbf{h}}_i$ are hidden states of forward and backward LSTMs respectively.

4 Training

We employ max-margin criterion [Taskar *et al.*, 2005] which provides an alternative to probabilistic based methods by optimizing on the robustness of decision boundary directly.

In the decoding phase, if the predicted tag sequence for the i-th training sentence $c^{(i)}_{1:n_i}$ with the maximal score is $\hat{t}^{(i)}_{1:n_i}$:

$$\hat{t}^{(i)}_{1:n_i} = \underset{t^{(i)}_{1:n_i} \in \mathbf{T}(c^{(i)}_{1:n_i})}{\arg\max} \mathbf{r}(t^{(i)}_{1:n_i}; \theta), \tag{14}$$

the goal of the max-margin criterion is to maximize the score of the gold tag sequence $t^{*(i)}_{1:n_i} = \hat{t}^{(i)}_{1:n_i}$ with a margin to any other possible tag sequence $t^{(i)}_{1:n_i} \in \mathbf{T}(c^{(i)}_{1:n_i})$:

$$\mathbf{r}(t^{*(i)}_{1:n_i}; \theta) \geq \mathbf{r}(t^{(i)}_{1:n_i}; \theta) + \Delta(t^{*(i)}_{1:n_i}, t^{(i)}_{1:n_i}), \tag{15}$$

$$\Delta(t^{*(i)}_{1:n_i}, t^{(i)}_{1:n_i}) = \sum_{j=1}^{n_i} \eta \mathbf{1}\{t^{*(i)}_j \neq t^{(i)}_j\}, \tag{16}$$

where $\Delta(t^{*(i)}_{1:n_i}, t^{(i)}_{1:n_i})$ is the margin function and hyper-parameter η is a discount parameter. Here, θ denotes all trainable parameters of our model.

Thus, the object is to minimize objective function $J(\theta)$ for m training examples $(c^{(i)}_{1:n_i}, t^{*(i)}_{1:n_i})^m_{i=1}$:

$$J(\theta) = \frac{1}{m} \sum_{i=1}^{m} l_i(\theta) + \frac{\lambda}{2} \|\theta\|_2^2, \tag{17}$$

$$l_i(\theta) = \max_{t^{(i)}_{1:n_i} \in \mathbf{T}(c^{(i)}_{1:n_i})} (\mathbf{r}(t^{(i)}_{1:n_i}; \theta) + \Delta(t^{*(i)}_{1:n_i}, t^{(i)}_{1:n_i})) - \mathbf{r}(t^{*(i)}_{1:n_i}; \theta). \tag{18}$$

Datasets	Splits	Splits	AVG$_w$	N_{sentence}	$\lvert\mathcal{D}_W\rvert$
CTB	Train	Sighan 2008	27.4	23,444	42k
	Test		28.8	2,079	10k
PKU	Train		16.7	66,691	55k
	Test		24.3	6,424	18k
NCC	Train		28.4	18,869	45k
	Test		28.5	3,595	18k
CTB-5	Train	1-270, 400-1151	27.3	18,086	37k
	Dev	301-325	19.4	350	2k
	Test	271-300	23.0	348	2k
CTB-7	Train	1-4197	24.0	41,266	52k
	Test	4198-4411	20.6	10,181	21k

Table 1: Details of five datasets. $\mathcal{D}_W$ is the dictionary of distinct words. N_{sentence} indicates the number of sentences. AVG$_w$ is the average word number in a sentence.

Window size	$k = 1$
Character embedding size	$d = 50$
Initial learning rate	$\alpha = 0.2$
Margin loss discount	$\eta = 0.2$
Regularization	$\lambda = 10^{-4}$
LSTM dimensionality	$h = 100$
Number of feature map sets	$Q = 5$
Size of each feature map set $\hat{f}^q$	$l^Q_{q=1} = 100$
Batch size	20

Table 2: Hyper-parameter settings.

5 Experiments

5.1 Datasets

We evaluate proposed architecture on five datasets: CTB, PKU, NCC, CTB-5, CTB-7. Table 1 gives the details of five datasets. We use the first 10% data of shuffled train set as development set for CTB, PKU, NCC and CTB-7 datasets.

- **CTB, PKU and NCC** datasets are from the POS tagging task of the Fourth International Chinese Language Processing Bakeoff [Jin and Chen, 2008].
- **CTB-5** dataset is the version of Penn Chinese Treebank 5.1, following the partition criterion of [Jin and Chen, 2008; Jiang *et al.*, 2009; Sun and Wan, 2012]
- **CTB-7** dataset is the version of Penn Chinese Treebank 7.0. It consists of different sources of documents (newswire, magazine articles, broadcast news, broadcast conversations, newsgroups and weblogs). Since the web blogs are very different with news texts, we try to evaluate the robustness of our model by testing on web blogs and training on the rest of dataset.

5.2 Hyper-parameters

Table 2 gives the details of hyper-parameter settings. Note that we set window size $k = 1$ which means we only take the current character embedding into account instead of using window slice approach. According to experiment results, we find it is a tradeoff between model performance and efficiency to only use { uni-gram, bi-gram, . . . , 5-gram } convolutional feature map sets. Besides, we set sizes of all feature map sets

models	w/o LSTM			LSTM			BLSTM		
	P	R	F	P	R	F	P	R	F
w/o CNN	-	-	-	89.24	89.66	89.45	89.52	89.74	89.63
CNN	88.24	89.16	88.70	89.35	89.71	89.53	89.75	89.61	89.68
CNN+Pooling	88.51	89.00	88.76	88.54	89.13	88.83	88.91	89.33	89.12
CNN+Pooling+Highway	**90.14**	**90.34**	**90.24**	**89.38**	**89.73**	**89.55**	**90.23**	**90.55**	**90.39**

Table 3: Performances of different models on test set of CTB dataset.

models	CTB			PKU			NCC			CTB5			CTB7		
	P	R	F	P	R	F	P	R	F	P	R	F	P	R	F
CRF	**90.51**	90.23	90.37	90.00	89.12	89.56	87.93	87.24	87.58	92.85	93.24	93.05	84.64	85.86	85.24
[Qiu *et al.*, 2013]	89.11	89.16	89.13	89.41	88.58	88.99	-	-	-	**93.28**	93.35	**93.31**	-	-	-
MLP	88.11	87.29	87.69	88.22	87.74	87.98	85.80	85.66	85.73	-	-	91.82*	83.60	84.53	84.06
Ours	89.48	89.63	89.56	89.82	89.55	89.68	87.30	87.76	87.53	91.78	92.88	92.33	84.02	**86.26**	85.13
Ours+Pre-train	90.23	**90.55**	**90.39**	**90.27**	**90.05**	**90.16**	**88.37**	**89.16**	**88.76**	92.88	**93.49**	93.19	**84.40**	86.25	**85.31**

Table 4: Comparisons with previous models on test sets of CTB, PKU, NCC, CTB5 and CTB7 datasets.

consistently for simplicity. Following previous work [Pei *et al.*, 2014; Chen *et al.*, 2015b], we also adopt bigram-character embedding in this paper.

5.3 Effects of Components

We experiment several models by using different neural component layers as shown in Table 3. The model incorporating convolutional layer, pooling layer, highway layer, and BLSTM layer, achieves the best performance on F1 score (90.39) on test set of CTB dataset. Therefore, we would like to compare our approach with other previous works using this topology.

Notably, the conventional model using window slice approach (Figure 2) for joint S&T task can be viewed as a special case of our model when we only adopt a singe convolutional layer.

Pooling Layer and Highway Layer

To evaluate the effectiveness of pooling layer and highway layer, we incrementally add pooling layer and highway layer on the top of convolutional layer. As shown in Table 3, by adding pooling layer, the performance decrease a little for the loss of information. However, we get the better performance on F1 score (90.24) by additionally adding highway layer. Although the performance does not benefit from pooling layer much, the pooling layer extracts the most important features and meets the consistent dimensionality requirement to add highway layer. Intuitively, highway layer helps simulating more complex compositional features by increasing the depth of architecture.

Long Short-Term Memory Layer

In this work, we introduce (B)LSTM layer to carry the long distance dependency. To evaluate (B)LSTM layer, we experiment different models with and without (B)LSTM layer. As shown in Table 3, we could get a relatively high performance by using LSTM or BLSTM layer only, which shows the capability of (B)LSTM in modeling features and carrying long distance information.

By introducing convolutional layer and highway layer, we could further boost the performance which benefits from the feature modeling capability of convolutional layer and highway layer.

5.4 Comparsion with Previous Works

We compare proposed model with several previous works on five datasets on joint S&T task. Experimental results are shown in Table 4.

Conditional random field (CRF) [Lafferty *et al.*, 2001] is one of the most prevalent and widely used models for sequence labeling tasks. [Qiu *et al.*, 2013] aims to boost the performance by exploiting datasets with different annotation types. Multilayer perceptron (MLP) is our implementation of [Zheng *et al.*, 2013], a basic neural model for joint S&T task. [Zheng *et al.*, 2013] is a neural model which only use one layer of shallow feed forward neural network in the encoding phase. Our model indicates the model with convolutional layer, pooling layer, highway layer, and BLSTM layer. "Pre-train" indicates the pre-trained character embeddings which are trained on corresponding train set of each dataset using word2vec toolkit [Mikolov *et al.*, 2013].

Result Discussion Our model outperforms the previous neural model on joint S&T task and achieves the comparable performance with conventional hand-crafted feature based models. As shown in Table 4, compared to other previous methods, our model achieves the best performances on F1 scores (90.39, 90.16, 88.76, 85.31 on CTB, PKU, NCC and CTB-7 datasets respectively), and obtains comparable results on CTB5 dataset (93.19 on F1 score). As we know, the test set of CTB5 is very small so that previous work might overfit on that dataset. In addition, according to the experimental results, we find that the performance benefits a lot from pre-trained character embeddings. Intuitively, pre-trained embeddings give a more reasonable initialization for the non-convex optimization problem with huge parameter space.

* [Zheng *et al.*, 2013] only reported the results on CTB5 dataset for joint S&T task.

Words	America 美国	knowledge 知识	information 信息	network 网络	company 公司	、	world 世界	study 学习	organization 组织	、	HaiSaiKe 海赛克	company 公司		
Tags	NR	NN	NN	NN	NN	PU	NN	~~VV~~ NN	NN	PU	NR	NN		
Words	incorporate 结合	reform 改革	、	simplify 精简	institution 机构	to 来	carry out 推行	civil servant 公务员	system 制度					
Tags	VV	~~VV~~ NN	PU	~~JJ~~ VV	NN	MSP	VV	NN	NN					
Words	via 通过	China 中	Portugal 葡	both sides 双方	's 的	friendly 友好	cooperation 合作	and 和	joint 共同	effort 努力				
Tags	P	NR	NR	PN	DEG	JJ	NN	CC	JJ	~~VV~~ NN				
Words	each 各	troop 部队	should 要	act for 为	reform 改革	、	development 发展	、	stability 稳定	offer 提供	reliably 可靠	's 的	security 安全	safeguard 保障
Tags	DT	NN	VV	P	NN	PU	NN	PU	~~AD~~ NN	VV	VA	DEC	NN	NN

Table 5: Case study. The red tags with strikethrough lines indicate the wrong predictions, which are the results of the CRF model. The green tags are corrected predictions made by the proposed feature-enriched neural model. The black tags are correctly tagged by all models.

Besides, the proposed model is quite efficient. It only takes about half one hour per epoch using a small amount of memory (to train CTB) on a single GPU. Actually, it takes about ten hours to train our model (on CTB).

Experiments on CTB-7, whose train set and test set are on different domains, show the robustness of our model.

5.5 Case Study

We illustrate several cases from CTB-5 dataset. As shown in Table 5, our approach performs well on cases with disambiguations which rely on long distance dependency. For instance, conditional random field (CRF) model makes mistakes on words " 改革" and "精简" since it is hard for CRF to disambiguate the POS tags without using the long distance (wider contextual) information.

6 Related Works

Recently, researches applied deep learning algorithms on various NLP tasks and achieved impressive results, such as chunking, POS tagging, named entity recognition for English [Collobert *et al.*, 2011; Tsuboi, 2014; Labeau *et al.*, 2015; Ma and Hovy, 2016; Santos and Zadrozny, 2014; Huang *et al.*, 2015], and Chinese word segmentation and POS tagging for Chinese [Zheng *et al.*, 2013; Pei *et al.*, 2014; Chen *et al.*, 2017]. These models learn features automatically which alleviate the efforts in feature engineering. However, joint S&T is a more difficult task than Chinese word segmentation and POS tagging since it has a larger decoding space and need more contextual information and long distance dependency [Zhang and Clark, 2008; Jiang *et al.*, 2008; Kruengkrai *et al.*, 2009; Zhang and Clark, 2010; Sun, 2011; Qian and Liu, 2012; Zheng *et al.*, 2013; Qiu *et al.*, 2013; Shen *et al.*, 2014]. Therefore, we need a customized architecture to alleviate these problems. In this work, we propose a feature-enriched neural model for joint S&T task, and obtain great performance.

Besides, there are several similar neural models [Tsuboi, 2014; Labeau *et al.*, 2015; Ma and Hovy, 2016; Santos and Zadrozny, 2014; Huang *et al.*, 2015; Kim *et al.*, 2015] . Instead of looking up word embedding table for each word in text, they tries to directly model English words by applying convolution layer on characters of words. Then they apply these word presentations to other tasks, such as POS tagging, name entity recognition, language modeling, etc. Unlike these models, we apply convolutional operation on sentence level, while they do within each word. Therefore they do not capture the features involving several words. Besides, we apply pooling operation along the feature size direction to get the most significant features.

7 Conclusions

In this paper, we propose a feature-enriched neural model for joint S&T task, which better models compositional features and utilizes long distance dependency. Experimental results show that our proposed model outperforms the previous neural model and achieves comparable results with previous sophisticated feature based approaches.

Acknowledgments

We would like to thank the anonymous reviewers for their valuable comments. This work is partially funded by National Natural Science Foundation of China (No. 61532011 and 61672162), Shanghai Municipal Science and Technology Commission on (No. 16JC1420401).

References

[Chen *et al.*, 2015a] Xinchi Chen, Xipeng Qiu, Chenxi Zhu, and Xuanjing Huang. Gated recursive neural network for chinese word segmentation. In *ACL*, 2015.

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

[Chen *et al.*, 2017] Xinchi Chen, Zhan Shi, Xipeng Qiu, and Xuanjing Huang. Adversarial multi-criteria learning for chinese word segmentation. In *ACL*, 2017.

[Collobert *et al.*, 2011] Ronan Collobert, Jason Weston, Léon Bottou, Michael Karlen, Koray Kavukcuoglu, and Pavel Kuksa. Natural language processing (almost) from scratch. *The Journal of Machine Learning Research*, 12:2493–2537, 2011.

[Elman, 1990] Jeffrey L Elman. Finding structure in time. *Cognitive science*, 14(2):179–211, 1990.

[Forney Jr, 1973] G David Forney Jr. The viterbi algorithm. *Proceedings of the IEEE*, 61(3):268–278, 1973.

[Hochreiter and Schmidhuber, 1997] Sepp Hochreiter and Jürgen Schmidhuber. Long short-term memory. *Neural computation*, 9(8):1735–1780, 1997.

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

[Jiang *et al.*, 2008] W. Jiang, L. Huang, Q. Liu, and Y. Lu. A cascaded linear model for joint Chinese word segmentation and part-of-speech tagging. In *ACL*. Citeseer, 2008.

[Jiang *et al.*, 2009] W. Jiang, L. Huang, and Q. Liu. Automatic adaptation of annotation standards: Chinese word segmentation and POS tagging: a case study. In *ACL*, pages 522–530, 2009.

[Jin and Chen, 2008] C. Jin and X. Chen. The fourth international chinese language processing bakeoff: Chinese word segmentation, named entity recognition and chinese pos tagging. In *Sixth SIGHAN Workshop on Chinese Language Processing*, page 69, 2008.

[Kim *et al.*, 2015] Yoon Kim, Yacine Jernite, David Sontag, and Alexander M Rush. Character-aware neural language models. *arXiv preprint arXiv:1508.06615*, 2015.

[Kruengkrai *et al.*, 2009] Canasai Kruengkrai, Kiyotaka Uchimoto, Jun'ichi Kazama, Yiou Wang, Kentaro Torisawa, and Hitoshi Isahara. An error-driven word-character hybrid model for joint chinese word segmentation and pos tagging. In *ACL*, pages 513–521. Association for Computational Linguistics, 2009.

[Labeau *et al.*, 2015] Matthieu Labeau, Kevin Löser, and Alexandre Allauzen. Non-lexical neural architecture for fine-grained pos tagging. In *EMNLP*, pages 232–237, Lisbon, Portugal, September 2015. Association for Computational Linguistics.

[Lafferty *et al.*, 2001] John D. Lafferty, Andrew McCallum, and Fernando C. N. Pereira. Conditional random fields: Probabilistic models for segmenting and labeling sequence data. In *Proceedings of the Eighteenth International Conference on Machine Learning*, 2001.

[Ma and Hovy, 2016] Xuezhe Ma and Eduard Hovy. End-to-end sequence labeling via bi-directional lstm-cnns-crf. *arXiv preprint arXiv:1603.01354*, 2016.

[Mikolov *et al.*, 2013] Tomas Mikolov, Kai Chen, Greg Corrado, and Jeffrey Dean. Efficient estimation of word representations in vector space. *arXiv preprint arXiv:1301.3781*, 2013.

[Pei *et al.*, 2014] Wenzhe Pei, Tao Ge, and Chang Baobao. Maxmargin tensor neural network for chinese word segmentation. In *Proceedings of ACL*, 2014.

[Qian and Liu, 2012] Xian Qian and Yang Liu. Joint chinese word segmentation, pos tagging and parsing. In *EMNLP*, pages 501–511. Association for Computational Linguistics, 2012.

[Qian *et al.*, 2010] Xian Qian, Qi Zhang, Yaqian Zhou, Xuanjing Huang, and Lide Wu. Joint training and decoding using virtual nodes for cascaded segmentation and tagging tasks. In *EMNLP*, 2010.

[Qiu *et al.*, 2013] Xipeng Qiu, Jiayi Zhao, and Xuanjing Huang. Joint chinese word segmentation and pos tagging on heterogeneous annotated corpora with multiple task learning. In *EMNLP*, pages 658–668, 2013.

[Santos and Zadrozny, 2014] Cicero D Santos and Bianca Zadrozny. Learning character-level representations for part-of-speech tagging. In *ICML*, pages 1818–1826, 2014.

[Shen *et al.*, 2014] Mo Shen, Hongxiao Liu, Daisuke Kawahara, and Sadao Kurohashi. Chinese morphological analysis with character-level pos tagging. In *ACL*, pages 253–258, 2014.

[Srivastava *et al.*, 2015] Rupesh Kumar Srivastava, Klaus Greff, and Jürgen Schmidhuber. Highway networks. *arXiv preprint arXiv:1505.00387*, 2015.

[Sun and Wan, 2012] Weiwei Sun and Xiaojun Wan. Reducing approximation and estimation errors for Chinese lexical processing with heterogeneous annotations. In *ACL*, pages 232–241, 2012.

[Sun, 2011] W. Sun. A stacked sub-word model for joint Chinese word segmentation and part-of-speech tagging. In *ACL*, pages 1385–1394, 2011.

[Taskar *et al.*, 2005] Ben Taskar, Vassil Chatalbashev, Daphne Koller, and Carlos Guestrin. Learning structured prediction models: A large margin approach. In *ICML*, 2005.

[Tsuboi, 2014] Yuta Tsuboi. Neural networks leverage corpus-wide information for part-of-speech tagging. In *EMNLP*, pages 938–950, Doha, Qatar, October 2014. Association for Computational Linguistics.

[Zhang and Clark, 2008] Yue Zhang and Stephen Clark. Joint word segmentation and pos tagging using a single perceptron. In *ACL*, pages 888–896, 2008.

[Zhang and Clark, 2010] Yue Zhang and Stephen Clark. A fast decoder for joint word segmentation and POS-tagging using a single discriminative model. In *EMNLP*, pages 843–852, Stroudsburg, PA, USA, 2010.

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