A Long Dependency Aware Deep Architecture for Joint Chinese Word Segmentation and POS 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

Long-term context is crucial to joint Chinese word segmentation and POS tagging (S&T) task. However, most of machine learning based methods extract features from a window of characters. Due to the limitation of window size, these methods can not exploit the long distance information. In this work, we propose a long dependency aware deep architecture for joint S&T 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. Experiment results on five different datasets show the effectiveness of our proposed model.

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. 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; Zhang and Clark 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 have millions of discrete features. Therefore, the efficiency of joint S&T models is relatively low. Moreover, these models suffer from data sparsity.

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 “(reform)” and “ (simplify)” since it is hard for CRF

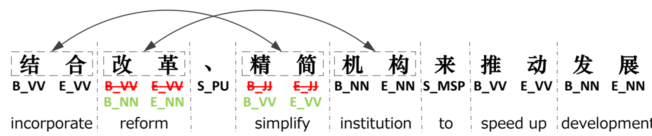

Figure 1: An example. CRF makes mistakes on words “改革(reform)” and “精(simplify)”. The red tags with strikethrough lines indicate the wrong predictions.

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 long dependency aware deep architecture, 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, Greff, and Schmidhuber 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 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 model.

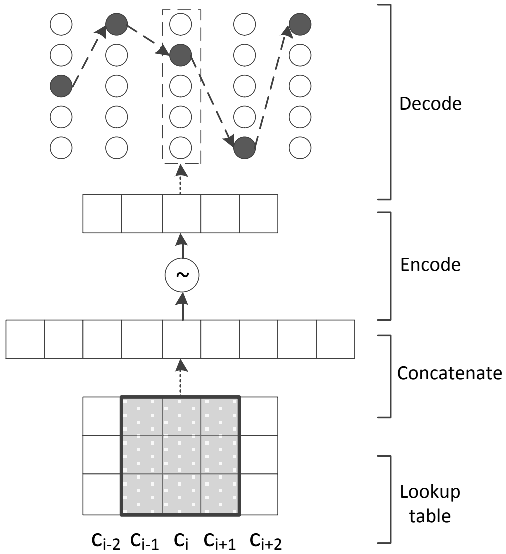

Figure 2: General neural network based architecture for joint S&T. The solid arrow denotes that there is a weight matrix on the link, while the dashed one denotes none.

Neural Models for Joint S&T

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

In the 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 “ ”, where $\text{B} \in \mathcal{T}_{SEG}$ and $\text{VV} \in \mathcal{T}_{POS}$, indicating the first character of the VV word “合”. Conventional neural network based model for sequence labeling task usually consists of three phases. Figure 2 gives the illustration.

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.

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 $\textbf{sigmoid}(\cdot)$, $\textbf{tanh}(\cdot)$, 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.

Decoding Phase

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

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

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

where $\mathbf{A} \in \mathbf{R}^{|\mathcal{T}|\times|\mathcal{T}|}$ is the transition parameter, and $\mathbf{T}(c_{1:n})$ 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.

A Long Dependency Aware Deep Architecture for Joint S&T

The simple neural model presented above achieves good results on the joint S&T task. However, since lots of ambiguous cases rely on long distance dependencies, a simple shallow neural is insufficient to capture long-distance information. In addition, the simple neural model concatenates the embeddings of contextual characters as features which is relative weaker than the hand-crafted features.

To deal with these issues, we propose a long dependency aware deep architecture for joint S&T, 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.

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

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

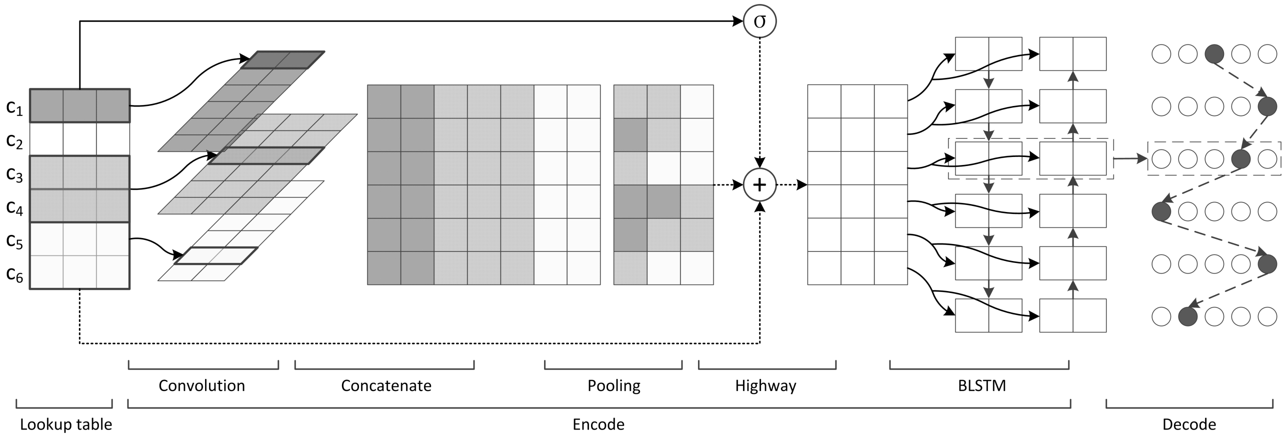

Figure 3: Proposed long dependency aware deep architecture for joint S&T. The solid arrow denotes that there is a weight matrix on the link, while the dashed one denotes none.

where $\mathbf{W}^q_{cov} \in \mathbf{R}^{qd \times l_q}$ 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 same length sequence as input by padding zeros to input.

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

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

where operator $\oplus$ is the concatenation operation.

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

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

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

Highway Layer

Highway layer (Srivastava, Greff, and Schmidhuber 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 input sequence as $\hat{\mathbf{X}} = Cov(\mathbf{X})$ after convolutional layer. By additionally adding highway layer, the representation of the input sequence would be $\hat{\mathbf{X}}$ as:

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

where operator $\odot$ indicates the element-wise multiplication operation. Here, $T(\cdot)$ 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:

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

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.

Recurrent Layer

Since that lots of cases with disambiguations rely on long distance dependency in joint S&T task, 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). LSTM 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. Hence, we would get $\mathbf{H} \in \mathbf{R}^{n \times 2h}$ as:

$$\mathbf{H} = \overrightarrow{\mathbf{H}} \oplus \overleftarrow{\mathbf{H}} \tag{13}$$

$$= \overrightarrow{\mathbf{LSTM}}(\hat{\mathbf{X}}) \oplus \overleftarrow{\mathbf{LSTM}}(\hat{\mathbf{X}}) \tag{14}$$

$$= \begin{bmatrix} \overrightarrow{\mathbf{h}}_1^\intercal \oplus \overleftarrow{\mathbf{h}}_1^\intercal \\ \overrightarrow{\mathbf{h}}_2^\intercal \oplus \overleftarrow{\mathbf{h}}_2^\intercal \\ \cdots \\ \overrightarrow{\mathbf{h}}_n^\intercal \oplus \overleftarrow{\mathbf{h}}_n^\intercal \end{bmatrix} \tag{15}$$

$$= [\mathbf{h}_1, \mathbf{h}_2, \ldots, \mathbf{h}_n]^\intercal, \tag{16}$$

where operator $\oplus$ indicates concatenation operation. Here, $\overrightarrow{\mathbf{LSTM}}(\cdot)$ is forward LSTM and $\overleftarrow{\mathbf{LSTM}}(\cdot)$ is backward LSTM.

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

$$\hat{t}^{(i)}_{1:n_i} = \operatorname*{arg\,max}_{t^{(i)}_{1:n_i} \in \mathbf{T}(c^{(i)}_{1:n_i})} \mathbf{r}(t^{(i)}_{1:n_i}; \theta), \tag{17}$$

the goal of the max-margin criterion is to maximize the 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 $t^{(i)}_{1:n_i} \in \mathbf{T}(c^{(i)}_{1:n_i})$:

$$\mathbf{r}({t^*}^{(i)}_{1:n_i}; \theta) \geq \mathbf{r}(t^{(i)}_{1:n_i}; \theta) + \Delta({t^*}^{(i)}_{1:n_i}, t^{(i)}_{1:n_i}), \tag{18}$$

$$\Delta({t^*}^{(i)}_{1:n_i}, t^{(i)}_{1:n_i}) = \sum_{j=1}^{n_i} \eta \mathbf{1}\{{t^*}^{(i)}_j \neq t^{(i)}_j\}, \tag{19}$$

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 $(c^{(i)}_{1:n_i}, {t^*}^{(i)}_{1:n_i})^m_{i=1}$:

$$J(\theta) = \frac{1}{m}\sum_{i=1}^{m} l_i(\theta) + \frac{\lambda}{2}\|\theta\|_2^2, \tag{20}$$

$$l_i(\theta) = \max_{t^{(i)}_{1:n_i} \in \mathbf{T}(c^{(i)}_{1:n_i})} (\mathbf{r}(t^{(i)}_{1:n_i}; \theta) + \Delta({t^*}^{(i)}_{1:n_i}, t^{(i)}_{1:n_i})) - \mathbf{r}({t^*}^{(i)}_{1:n_i}; \theta). \tag{21}$$

Experiments

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.

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 distinguished 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.

- **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, Huang, and Liu 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.

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 consistently for simplicity.

Model Selection

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

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		
	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
(Qiu, Zhao, and Huang 2013)	89.11	89.16	89.13	89.41	88.58	88.99	-	-	-
MLP	88.11	87.29	87.69	88.22	87.74	87.98	85.80	85.66	85.73
LD-DNN*	89.48	89.63	89.56	89.82	89.55	89.68	87.30	87.76	87.53
LD-DNN	90.23	**90.55**	**90.39**	**90.27**	**90.05**	**90.16**	**88.37**	**89.16**	**88.76**

Table 4: Comparisons with previous models on test sets of CTB, PKU, NCC datasets (SIGHAN 2008).

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 (LD-DNN).

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 LD-DNN only uses 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.

Comparsion with Previous Works

We compare proposed model LD-DNN with several previous works on five datasets on joint S&T task. Experiment results are shown in Table 4, Table 5 and Table 7 respectively.

models	CTB5		
	P	R	F
CRF	92.85	93.24	93.05
(Zhang and Clark 2008)	-	-	91.34
(Jiang et al. 2008)	-	-	93.41
(Kruengkrai et al. 2009)	**94.07**	93.28	93.67
(Zhang and Clark 2010)	-	-	93.67
(Sun 2011)	-	-	94.02
(Qian and Liu 2012)	93.10	93.96	93.53
(Qiu, Zhao, and Huang 2013)	93.28	93.35	93.31
(Shen et al. 2014)	93.42	94.18	93.80
(Zheng, Chen, and Xu 2013)	-	-	91.82
LD-DNN*	92.28	93.38	92.83
LD-DNN	93.88	**94.21**	**94.04**

Table 5: Comparisons with previous models on test set of CTB-5 dataset.

models	CTB7		
	P	R	F
CRF	84.64	85.86	85.24
MLP	83.60	84.53	84.06
LD-DNN*	84.02	**86.26**	85.13
LD-DNN	**84.40**	86.25	**85.31**

Table 7: Comparisons with previous models on test set of CTB-7 dataset.

Words	incorporate 合	reform 改革	、	simplify 精	institution 机构	to 来	carry out 推行	civil servant 公	system 制度					
Tags	VV	~~VV~~ NN	PU	~~JJ~~ VV	NN	MSP	VV	NN	NN					
Words	via 通	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	should 要	act for	reform 改革	、	development 展	、	stability 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 6: Case study. The red tags with strikethrough lines indicate the wrong predictions, which are the results of the models without using LSTM neural networks like CRF. The green tags are corrected predictions made by models using LSTM like our proposed LD-DNN. The black tags are correctly tagged by all models.

Conditional random field (CRF) (Lafferty, McCallum, and Pereira 2001) is one of the most prevalent and widely used models for sequence labeling tasks. (Zheng, Chen, and Xu 2013) is a neural model which only use one layer of shallow feed forward neural network in their encoding phase. Multilayer perceptron (MLP) is our implementation of (Zheng, Chen, and Xu 2013), a basic neural model for joint S&T task. (Qiu, Zhao, and Huang 2013) aims to boost the performance by exploiting datasets with different annotation types. (Zhang and Clark 2008; Jiang et al. 2008; Kruengkrai et al. 2009; Zhang and Clark 2010; Sun 2011; Qian and Liu 2012; Shen et al. 2014) are conventional feature based models which exploit well designed sophisticated features. Besides, we experiment proposed model LD-DNN with and without using pre-trained character embeddings.

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, Table 5 and Table 7, compared to other previous methods, our model LD-DNN achieves best performances on F1 scores (90.39, 90.16, 88.76, 94.04 and 85.31 on CTB, PKU, NCC, CTB-5 and CTB-7 datasets respectively) on all datasets.

Besides, proposed LD-DNN is quite efficient. LD-DNN 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 LD-DNN (on CTB).

In addition, according to the experiment results, we find that 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.

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

Case Study

We illustrate several cases from CTB-5 dataset. As shown in Table 6, our architecture with LSTM layer 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" since it is hard for CRF to disambiguate the POS tags without using the long distance (wider contextural) information.

LD-DNN* indicates our model without using pre-trained character embeddings.

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, Löser, and Allauzen 2015; Ma and Hovy 2016; Santos and Zadrozny 2014; Huang, Xu, and Yu 2015), and Chinese word segmentation and POS tagging for Chinese (Zheng, Chen, and Xu 2013; Pei, Ge, and Baobao 2014; Chen et al. 2015). 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. Therefore, we need a customized architecture to alleviate these problems. In this work, we propose a long dependency aware deep architecture for joint S&T task, and obtain great performance.

Besides, there are several similar neural models (Tsuboi 2014; Labeau, Löser, and Allauzen 2015; Ma and Hovy 2016; Santos and Zadrozny 2014; Huang, Xu, and Yu 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.

Conclusions

In this paper, we propose a long dependency aware deep architecture 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.

References

[Chen et al. 2015] Chen, X.; Qiu, X.; Zhu, C.; and Huang, X. 2015. Gated recursive neural network for chinese word segmentation. In In Proceedings of the Conference on Empirical Methods in Natural Language Processing.

[Collobert et al. 2011] Collobert, R.; Weston, J.; Bottou, L.; Karlen, M.; Kavukcuoglu, K.; and Kuksa, P. 2011. Natural language processing (almost) from scratch. The Journal of Machine Learning Research 12:2493–2537.

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

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

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

[Huang, Xu, and Yu 2015] Huang, Z.; Xu, W.; and Yu, K. 2015. Bidirectional lstm-crf models for sequence tagging. arXiv preprint arXiv:1508.01991.

[Jiang et al. 2008] Jiang, W.; Huang, L.; Liu, Q.; and Lu, Y. 2008. A cascaded linear model for joint Chinese word segmentation and part-of-speech tagging. In In Proceedings of the 46th Annual Meeting of the Association for Computational Linguistics. Citeseer.

[Jiang, Huang, and Liu 2009] Jiang, W.; Huang, L.; and Liu, Q. 2009. Automatic adaptation of annotation standards: Chinese word segmentation and POS tagging: a case study. In Proceedings of the Joint Conference of the 47th Annual Meeting of the ACL and the 4th International Joint Conference on Natural Language Processing, 522–530.

[Jin and Chen 2008] Jin, C., and Chen, X. 2008. 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, 69.

[Kim et al. 2015] Kim, Y.; Jernite, Y.; Sontag, D.; and Rush, A. M. 2015. Character-aware neural language models. arXiv preprint arXiv:1508.06615.

[Kruengkrai et al. 2009] Kruengkrai, C.; Uchimoto, K.; Kazama, J.; Wang, Y.; Torisawa, K.; and Isahara, H. 2009. An error-driven word-character hybrid model for joint chinese word segmentation and pos tagging. In Proceedings of the Joint Conference of the 47th Annual Meeting of the ACL and the 4th International Joint Conference on Natural Language Processing of the AFNLP: Volume 1-Volume 1, 513–521. Association for Computational Linguistics.

[Labeau, Löser, and Allauzen 2015] Labeau, M.; Löser, K.; and Allauzen, A. 2015. Non-lexical neural architecture for fine-grained pos tagging. In Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing, 232–237. Lisbon, Portugal: Association for Computational Linguistics.

[Lafferty, McCallum, and Pereira 2001] Lafferty, J. D.; McCallum, A.; and Pereira, F. C. N. 2001. Conditional random fields: Probabilistic models for segmenting and labeling sequence data. In Proceedings of the Eighteenth International Conference on Machine Learning.

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

[Pei, Ge, and Baobao 2014] Pei, W.; Ge, T.; and Baobao, C. 2014. Maxmargin tensor neural network for chinese word segmentation. In Proceedings of ACL.

[Qian and Liu 2012] Qian, X., and Liu, Y. 2012. Joint chinese word segmentation, pos tagging and parsing. In Proceedings of the 2012 Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning, 501–511. Association for Computational Linguistics.

[Qian et al. 2010] Qian, X.; Zhang, Q.; Zhou, Y.; Huang, X.; and Wu, L. 2010. Joint training and decoding using virtual nodes for cascaded segmentation and tagging tasks. In Proceedings of the 2010 Conference on Empirical Methods in Natural Language Processing.

[Qiu, Zhao, and Huang 2013] Qiu, X.; Zhao, J.; and Huang, X. 2013. Joint chinese word segmentation and pos tagging on heterogeneous annotated corpora with multiple task learning. In EMNLP, 658–668.

[Santos and Zadrozny 2014] Santos, C. D., and Zadrozny, B. 2014. Learning character-level representations for part-of-speech tagging. In Proceedings of the 31st International Conference on Machine Learning (ICML-14), 1818–1826.

[Shen et al. 2014] Shen, M.; Liu, H.; Kawahara, D.; and Kurohashi, S. 2014. Chinese morphological analysis with character-level pos tagging. In ACL (2), 253–258.

[Srivastava, Greff, and Schmidhuber 2015] Srivastava, R. K.; Greff, K.; and Schmidhuber, J. 2015. Highway networks. arXiv preprint arXiv:1505.00387.

[Sun and Wan 2012] Sun, W., and Wan, X. 2012. Reducing approximation and estimation errors for Chinese lexical processing with heterogeneous annotations. In Proceedings of the 50th Annual Meeting of the Association for Computational Linguistics, 232–241.

[Sun 2011] Sun, W. 2011. A stacked sub-word model for joint Chinese word segmentation and part-of-speech tagging. In Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies, 1385–1394.

[Taskar et al. 2005] Taskar, B.; Chatalbashev, V.; Koller, D.; and Guestrin, C. 2005. Learning structured prediction models: A large margin approach. In Proceedings of the international conference on Machine learning.

[Tsuboi 2014] Tsuboi, Y. 2014. Neural networks leverage corpus-wide information for part-of-speech tagging. In

Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP), 938–950. Doha, Qatar: Association for Computational Linguistics.

[Zhang and Clark 2008] Zhang, Y., and Clark, S. 2008. Joint word segmentation and pos tagging using a single perceptron. In ACL, 888–896.

[Zhang and Clark 2010] Zhang, Y., and Clark, S. 2010. A fast decoder for joint word segmentation and POS-tagging using a single discriminative model. In Proceedings of the 2010 Conference on Empirical Methods in Natural Language Processing, 843–852.

[Zheng, Chen, and Xu 2013] Zheng, X.; Chen, H.; and Xu, T. 2013. Deep learning for chinese word segmentation and pos tagging. In EMNLP, 647–657.