Research Paper

DOI: 10.3969/j. issn. 1673-5188. 2017. S2. 010
http://kns.cnki.net/kcms/detail/34.1294.TN.20171208.1121.002.html, published online December 8, 2017

Deep Neural Network-Based Chinese Semantic Role Labeling

ZHENG Xiaoqing[1], CHEN Jun[2], and SHANG Guoqiang[2]
(1. School of Computer Science, Fudan University, Shanghai 201203, China;
2. Terminal Business Division, ZTE Corporation, Shanghai 201203, China)

Abstract

A recent trend in machine learning is to use deep architectures to discover multiple levels of features from data, which has achieved impressive results on various natural language processing (NLP) tasks. We propose a deep neural network-based solution to Chinese semantic role labeling (SRL) with its application on message analysis. The solution adopts a six-step strategy: text normalization, named entity recognition (NER), Chinese word segmentation and part-of-speech (POS) tagging, theme classification, SRL, and slot filling. For each step, a novel deep neural network-based model is designed and optimized, particularly for smart phone applications. Experiment results on all the NLP sub-tasks of the solution show that the proposed neural networks achieve state-of-the-art performance with the minimal computational cost. The speed advantage of deep neural networks makes them more competitive for large-scale applications or applications requiring real-time response, highlighting the potential of the proposed solution for practical NLP systems.

Keywords

deep learning; sequence labeling; natural language understanding; convolutional neural network; recurrent neural network

1 Introduction

The goal of semantic role labeling (SRL) is to identify arguments for a predicate and assign semantically meaningful labels to them. A semantic role defines a semantic relation between a predicate and one of its arguments [1]. Typical roles include agent, patient, source, goal, and so on, which are core to a predicate, as well as time, location, manner, cause, and so forth, which are peripheral. Such semantic information is important in answering who, what, where, when, and why questions, which therefore is critical to high-level natural language processing (NLP) tasks such as information extraction [2], summarization [3], machine translations [4], and question-answering [5].

There is considerable work on English SRL, and the performance using automatic parses on the Penn Treebank has approached 0.81 F-score [6]. Research on Chinese SRL is still in its infancy. Although most of the machine learning techniques used in English SRL can be readily transferable to Chinese, there are a few inherent linguistic properties of Chinese that make syntactic parsing a particularly challenging task, not to mention semantic-level tasks. With the fact that Chinese texts are written without using whitespace to delimit words , the parsers have to build structures from characters rather than words. The second has to deal with is that Chinese with its (almost) complete lack of morphological marking for parts of speech certainly exhibits a higher degree of polysemy than English [7], which makes it hard for the parsers to decide the part-of-speech (POS) tags of the words.

Chinese SRL, however, is one of the most fundamental NLP tasks because of its importance in mediating between linguistic meaning and expression. Chinese SRL has received steady attention since the release of Chinese PropBank [8]. Most of the previous work focused on finding relevant features for the model component, and on finding effective statistical techniques for parameter estimation. Although such performance improvements can be useful in practice, there is a great temptation to optimize the performance of a system for a specific benchmark. Furthermore, such systems, especially for those that use joint solution, usually involve a great number of features, which makes engineering effective task-specific features and structural learning of parameters very hard.

Deep learning algorithm emerged as a successful machine learning technique five years ago. With the deep architectures, it became possible to learn high-level (compact) representations, each of which combines features at lower levels in an exponential and hierarchical way [9]. Unlike traditional linear statistical models such as conditional random fields (CRFs), those feature representations can be automatically discovered to be relevant to the task of interest, and thus the tasks-specific engineering could be avoided.

2 Problem and Solution

In this study, SRL is formulated as assigning (task-specific) labels to words of an input sentence. Once each word of the sentence is associated with a label from a set of pre-defined tags, those word-label pairs can be used to fill the slots of semantic templates or frames, which might be further transformed into a query to knowledge bases or databases. For ex-

This work was supported in part by a grant from ZTE Research Funding, and in part by a grant from Shanghai Municipal Natural Science Foundation (No. 13ZR1403800, No. 15511104303).

ample, we receive the following message:

明天上午九点在第一会议室我们与技术部开会讨论项目进展。

"We will have a meeting on the progress of the project with the colleagues from the technology section in the meeting room No. 1, 9 am tomorrow."

The input sentences are first classified into one of the categories, and each category represents a theme (or event) of interest. Note that different semantic labels can be defined for different themes. For the aforementioned example, its theme is recognized as "meeting", and the corresponding semantic labels include topic, date, time, location, people, and so on. Uninterested or irrelevant words in the sentence are normally labeled with a special label "O". The SRL results are listed as shown in **Table 1**.

Some necessary pre-processing steps are required to perform SRL task for Chinese, including text normalization, word segmentation, POS tagging, and named entity recognition (NER). As shown in **Fig. 1**, we propose an architecture of Chinese SRL system to integrate multiple modules required for Chinese SRL.

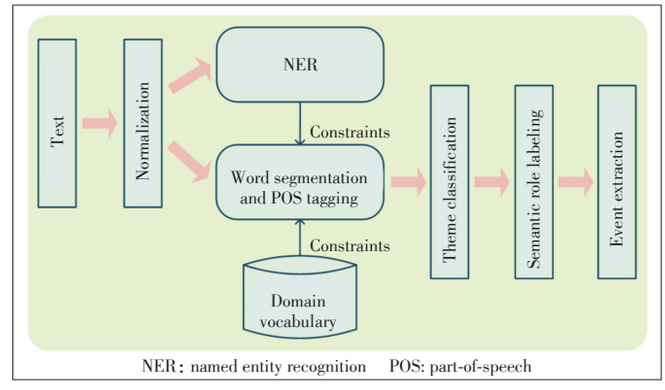
The proposed system works by the following steps.

- The system takes a Chinese sentence as input.
- The input sentence will be transformed into a standard form by transitioning from any other to UTF-8 encoding, from full-width symbols to half-width characters, from traditional Chinese characters to simplified ones, and replacing dialect, slang and informal expressions with formal words.
- Named entity recognition is performed over the sentence by a sequence labeling module based on neural networks. This module is used to identify person names, organizations, and locations.
- Several kinds of the pre-defined words (usually application-dependent) are identified from the input sentence. Such words are used to describe the names of food, idioms, devices, etc. Email addresses, dates, times, punctuations, uniform/universal resource locators, phone numbers, percentages, foreign words, figures, and currencies are also automatically recognized.
- Word segmentation and POS-tagging are performed in a joint way by neural networks using the same architecture designed for named entity recognition. The boundaries of the words recognized in the previous steps will be taken as constraints for this module.
- A convolutional neural network with multiple dynamic k-max pooling layers is then used to classify the sentence into one of several pre-defined themes. A set of semantic labels that may be used as semantic roles in the next step are also determined. If the input sentence does not belong to any theme of interest, it will be discard and does not need further processing.
- Each word will be assigned with an appropriate semantic label using bidirectional long short-term memory (bi-LSTM). The long short-term memory (LSTM) is a widely used variant of recurrent neural networks. For each theme, a bi-LSTM will be trained.
- The associated word-label pairs are extracted to fill the slots of semantic frame. Frame is a classic knowledge representation (KR) method [10], which has been successfully used for many intelligent systems. We, here, use the event extraction as sample application although the proposed architecture can be used in many ways.

We will describe how to perform the aforementioned modules in the next section. A neural network-based model was designed for each core function of the solution. The speed advantage of the neural networks makes them more competitive for large-scale applications requiring real-time response.

▼**Table 1. The SRL results**

Words	Labels
明天 "tomorrow"	DATE
上午九点 "9 am"	TIME
在 "at"	O
第一会议室 "meeting room No. 1"	LOCATION
我们 "we"	O
与 "with"	O
技术部 "technology section"	PARTICIPANT
开会 "have a meeting"	O
讨论 "discuss"	O
项目进展 "the progress of the project"	TOPIC
。"."	O



▲**Figure 1. Architecture of Chinese SRL system.**

3 Deep Neural Network-Based Models

3.1 Sequence Labeling

Word segmentation, POS tagging and named entity recognition all can be viewed as assigning labels to units of an input sentence. Many Asian languages, such as Chinese and Japanese, are written without whitespaces indicating the word

boundaries. For those languages, the character becomes a more natural form of input, and the labeling can be performed at character level. For Chinese word segmentation, each character will be assigned with one of four possible boundary tags: "*B*" for a character located at the beginning of a word, "*I*" for that inside of a word, "*E*" for that at the end of a word, and "*S*" for a character that is a word by itself. Following Ng and Lou [11] we perform joint word segmentation and POS tagging and NER in a labeling fashion by expanding boundary labels to include task-specific tags. Taking POS tagging as example, we describe noun phases using four different tags. A tag "*S_NP*" is used to mark a noun phase containing a single character. Other tags "*B_NP*", "*I_NP*", and "*E_NP*" are used to mark the first, in-between and the last characters of the noun phrase.

For sequence labeling tasks, we choose to use a variant of neural network architecture first introduced by Collobert et al [12] for multiple NLP tasks, and reintroduced later by Zheng et al [13] for joint word segmentation and POS tagging. However, a different algorithm is used to train the networks, and the boundaries of the words recognized in other modules will be taken as constraints for label inference.

The network architecture is shown in **Fig. 2**. The first layer extracts features for each character. The next layer extracts features from a window of characters. The following layers are classical neural network layers. The output is a graph over which tag inference is achieved with Viteri algorithm.

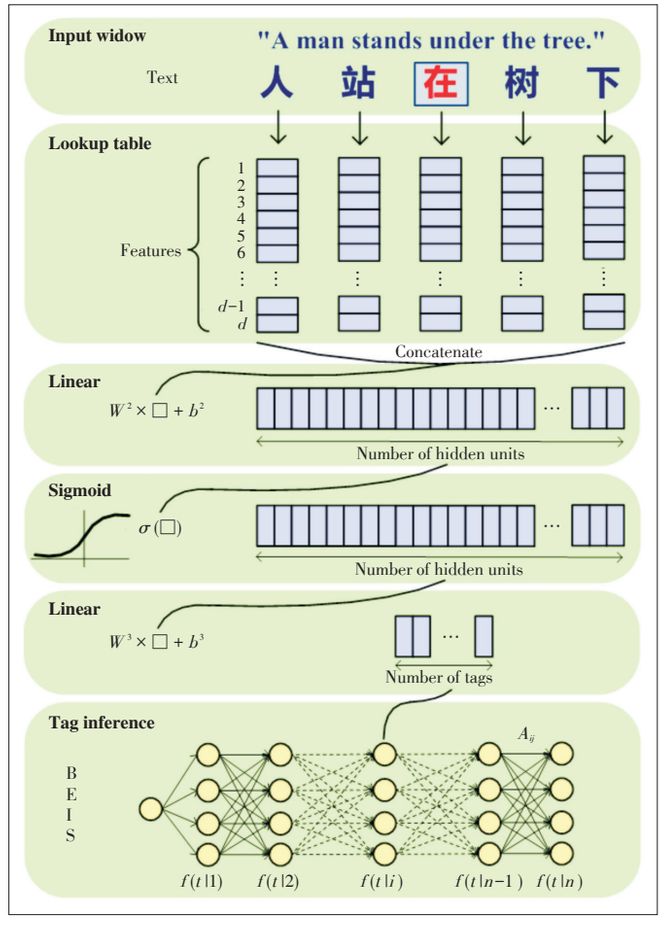
3.1.1 Mapping Characters into Feature Vectors

The characters are fed into the network as indices that are used by a lookup operation to transform tokens into their feature vectors. We consider a fixed-sized dictionary D (Unknown characters are mapped to a special symbol that is not used elsewhere). The feature vectors are stored in a character embedding matrix $M \in \mathbb{R}d \times |D|$, where d is the dimensionality of the vector space (a hyper-parameter) and $|D|$ is the size of the dictionary.

Formally, we assume that a sentence $x[1{:}n]$ is a sequence of n characters x_i, $1 \leqslant i \leqslant n$. For each token $x_i \in D$ that has an associated index e_i into the column of the embedding matrix, a d-dimensional feature vector is retrieved by the lookup table layer $G(x_i) = Mvei$, where we use a binary vector $vei \in \mathbb{R}|D| \times 1$ which is zero in all positions except at the e_i-th index. The feature vector of each character can be initialized at random or pre-trained on unlabeled corpora, which will be fined-tuned to be relevant to the task by back-propagating errors.

3.1.2 Label Scoring and Inference

For each character in a sentence, a score is produced for each label by applying several layers of the neural network over feature vectors produced by the lookup table. Given an input sentence $x[1{:}n]$, we consider all successive windows of size w, sliding over the sentence, from character x_1 to x_n. The characters with indices exceeding the sentence boundaries are



▲**Figure 2. The neural network architecture for sequence labeling tasks.**

mapped to one of the two special symbols, namely "*start*" and "*stop*" symbols.

Generally, there are strong dependencies between the labels for the sequence labeling tasks, such as word segmentation, NER and POS tagging. The labels are usually organized in chunks, and it is impossible for some labels to follow a particular label. Given a set of labels T for the task of interest, we introduce a transition score $t(j|i)$ for jumping from ith to jth label in successive characters, and an initial scores $t(j|\circ)$ for starting from the jth label, where $i, j \in T$. We want valid label paths to be encouraged, while discouraging all the others.

Given an input sentence $x_{[1:n]}$, the network outputs the matrix of scores $f_\theta(x_{[1:n]})$, in which an element of the column i is denoted by $f_\theta(j|i,k)$, indicating the score output by the network with parameters θ, for the jth label and for the previous label k, at the position i. The score of a sentence $x_{[1:n]}$ along a path of labels $y_{[1:n]}$ is then given by the sum of transition and network scores:

$$s(x_{[1,n]}, y_{[1,n]}, \theta) = \sum_{i=1}^{n} (t(y_i|y_{i-1}) + f_\theta(y_i|x_i, y_{i-1})) . \quad (1)$$

Given a sentence $x_{[1:n]}$, we can find the optimal label path $y^*_{[1:}$

$_{n]}$ by maximizing the sentence score:

$$y^*_{[1:n]} = \arg\max_{\forall y_{[1:n]}} s(x_{[1:n]}, y_{[1:n]}, \theta) . \quad (2)$$

The Viterbi algorithm is the natural choice for this inference.

3.1.3 Training

Given a training example (x, y), we define a structured margin $\Delta(x, y, \hat{y})$ loss for proposing a path $\hat{y}$ for sentence x when y is the true path. We drop the subscript $[1: n]$ from now for notation simplification. This penalty is proportional to the number of labels on which the two paths do not agree. In general, $\Delta(x, y, \hat{y})$ is equal to 0 if $y = \hat{y}$. The loss function is defined as a penalization of incorrect paths, where κ is a penalization term to incorrect labels.

$$\Delta(x, y, \hat{y}) = \sum_{i=1}^{n} \kappa 1\{\hat{y}_i \neq y_i\} . \quad (3)$$

For a training set, we seek to determine all the parameters θ of the network with small expected loss on unseen sentences. The score of a path $\hat{y}$ is higher if the algorithm is more confident that the path $\hat{y}$ is correct. In the max-margin estimation framework, we want to ensure that the highest scoring path is the true one for all training instances (x^i, y^i), $i = 1, \cdots, n$, and its score can be larger up to a margin defined by the loss. For all i in the training data:

$$s(x^i, y^i, \theta) \geq s(x^i, \hat{y}, \theta) + \Delta(x^i, y^i, \hat{y}) . \quad (4)$$

These lead us to minimize the following regularized objective for n training instances:

$$J(\theta) = \frac{1}{n}\sum_{i=1}^{n} E^i(\theta) + \frac{\lambda}{2}||\theta||^2 ,$$
$$E^i(\theta) = \max_{\hat{y}} (s(x^i, \hat{y}, \theta) + \Delta(x^i, y^i, \hat{y})) - s(x^i, y^i, \theta) , \quad (5)$$

where λ governs the relative importance of the regularization term compared with the error. The loss penalizes paths more when they deviate from the correct one. Minimizing this objective maximizes the score of the correct path, and minimizes that of the highest scoring but incorrect one. The objective is not differentiable due to the hinge loss. The subgradient method is used to compute a gradient-like direction.

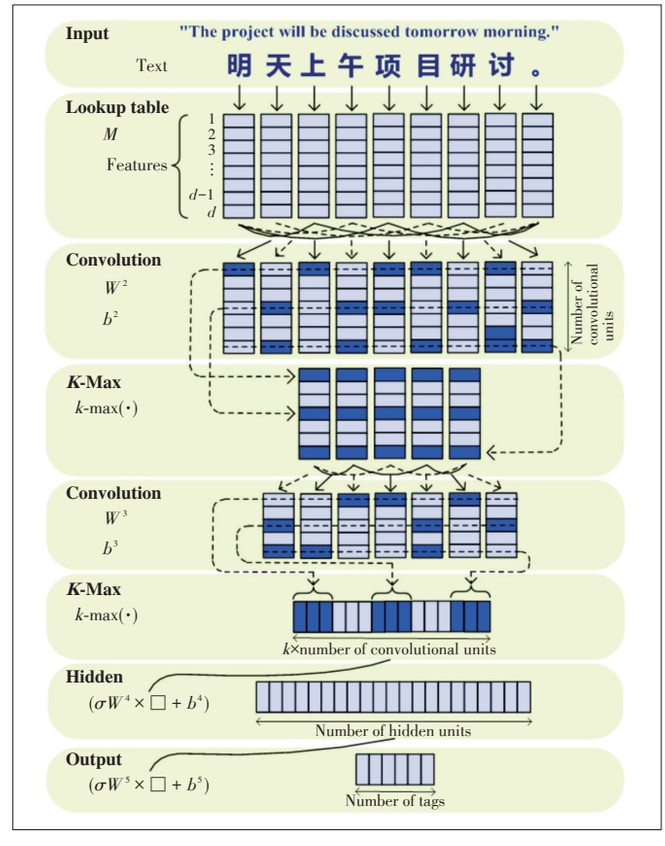
3.2 Sentence Classification

We choose to use a variant of convolutional neural network with dynamic k-max pooling [14] to find the structure of sentences. The network architecture is shown in **Fig. 3**. The k-max pooli ng operations are applied in the network after the convolutional layers, which are used to pool the k most active features at low levels. This network preserves the relative positions of the most relevant features, and it is sensitive to the order of the characters in the input sentences. Convolutional layers are often stacked, interleaved with a non-linearity function, to extract higher level features (only two convolutional layers are drawn in Fig. 3 for clarity). The topmost k-max pooling layer also guarantees that the input to the next layer of the network is independent of the length of an input sentence, in order to apply the same subsequent layers.

3.2.1 Convolutional Layer

The characters are fed into the network as indices that are used by a lookup operation (see Section 3.1.1) to transform characters into their feature vectors. The lookup table layer extracts features for each single character, but the features of a character in context will be influenced by surrounding characters. We assume that the features of a particular character depend mainly on its neighboring characters, and extract these features from a fixed size window w (a hyper-parameter). More precisely, given an input sentence $x_{[1:n]}$, the character feature window produced by the first lookup table layer at position x_i can be written as:

$$f_\theta^{win}(x_i) = \begin{pmatrix} G(x_{i-w/2}) \\ \vdots \\ G(x_i) \\ \vdots \\ G(x_{i+w/2}) \end{pmatrix} , \quad (6)$$



▲**Figure 3. The neural network architecture for sentence classification.**

where f_θ^{win} is a function with trainable parameters θ. The idea behind the convolution is to perform affine transformations over each character window to extract local features:

$$f_\theta^{con}(x_i)=(Wf_\theta^{win}(x_i)+b), \quad (7)$$

where the matrices $W \in \mathbb{R}V \times (wd)$ and $b \in \mathbb{R}V$ are the parameters to be optimized by training. A hyper-parameter V is called as the number of convolutional units. The trained weights in W and b can be viewed as a linguistic feature detector that learns to recognize a specific class of n-grams.

3.2.2 k-Max Pooling Layer

We applied a k-max pooling operation to the output of the convolutional layer, which is a generalization of max pooling over time dimension. Given a number k and a sequence $Q \in \mathbb{R}^p$ ($k \leqslant p$), k-max pooling selects the subsequence $Q^k{}_{max}$ of k-highest values in Q. The selected values of $Q^k{}_{max}$ preserve their relative order in Q. Given an output matrix f_θ^{con}, the k-max pooling layer yields another matrix:

$$f_\theta^{\max}(f_\theta^{con})=\begin{pmatrix} k\max([f_\theta^{con}]_{1,1} & \cdots & [f_\theta^{con}]_{1,j-i+w}) \\ \vdots & \vdots & \vdots \\ k\max([f_\theta^{con}]_{V,1} & \cdots & [f_\theta^{con}]_{V,j-i+w}) \end{pmatrix}, \quad (8)$$

where $[f_\theta^{con}]_{ij}$ is the element in the i-th row and j-th column of matrix f_θ^{con}.

Multiple convolutional layers are often stacked to extract higher level features. Different convolutional layers may have their own window size w and value k. Every column vector of the matrix produced by the topmost k-max pooling layer is concatenated to be fed to further neural network layers. For each layer, we let k be a function of the length of the input sentence and the depth of the network. We simply model $k_l = \max(k_{top}, (L - l) / L \times S)$ for lth layer, where L is the total number of convolutional layers, and k_{top} is the fixed pooling parameter for the topmost layer.

3.2.3 Classification with Softmax Layer

The fixed-sized vector produced by the topmost k-max pooling layer is fed to two standard linear layers that successively perform affine transformations over the vector, interleaved with some non-linearity function $\sigma(\cdot)$, to extract highly non-linear features. As for non-linear function, we choose the sigmoidal function.

We add a simple softmax layer to the network to predict theme categories for each input sentence. The classification is trained by minimizing the cross-entropy error of the softmax layer using backpropagation. When minimizing the cross-entropy error of the softmax layer, the error will also be propagated back and influence both the network parameters and the character representations.

3.3 Semantic Role Labeling

Chinese SRL is still an especially challenging task due to the long-range dependency phenomenon that arises when trying to assign semantically meaningful labels to each word for Chinese sentences. It relates to the rate of decay of the statistical dependence of two words with increasing the spatial distance between them. Work on Chinese SRL has been scant and sporadic [1]. We propose a variant of bidirectional long short-term memory to perform Chinese SRL for its ability to bridge long time lags between relevant inputs [15],[16].

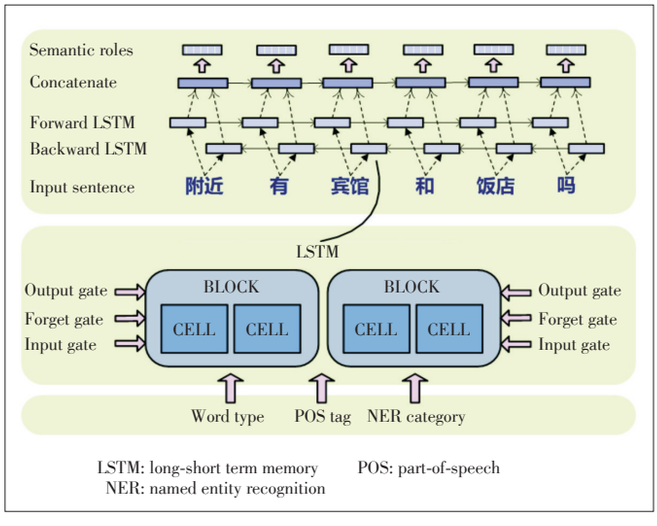
3.3.1 Architecture of Bidirectional LSTM

Hochreiter and Schmidhuber proposed a type of recurrent neural networks (RNN) called LSTM that works better than traditional RNNs on tasks involving long time lags [15]. Its architecture permits long short-term memory to bridge huge time lags between relevant input events, while traditional RNNs using more costly update algorithms. In Chinese SRL, we want to access both past and future input features for a given time, and use a variant of bidirectional LSTM (**Fig. 4**), first introduced by Graves et al[17].

The forward and backward processing over the unfolded network over time are done in a similar way to regular network forward and backward passes. For each time, the input to the network is the concatenation of word embedding, vector representation of the corresponding POS tag, and feature vector of the NER category. Word segmentation, POS tagging, and NER will be performed in advance, and their results will be used to generate the input vector for each word of the input sentence. The semantic labels will be predicted over the concatenation of network outputs produced by the forward and backward passes using softmax layer.

3.3.2 Training

The training problem is to determine all the parameters of the network from training data. Generally, the network is trained by maximizing the likelihood over all the sentences in



▲**Figure 4. The bi-directional long short-term memory for Chinese SRL.**

the training set R with respect to the parameters θ.

$$\theta \mapsto \sum_{\forall(x,y)\in R} \log p(y|x,\theta), \tag{9}$$

where x represents a sentence and its associated feature, and y denotes the corresponding label sequence. The probability $p(\cdot)$ is calculated from the outputs of the neural network.

Maximizing the log-likelihood with the gradient ascent algorithm is achieved by iteratively selecting an example (x, y) and applying the following gradient update rule:

$$\theta \leftarrow \theta + \alpha \frac{\partial \log p(y|x,\theta)}{\partial \theta}, \tag{10}$$

where α is the learning rate (a hyper-parameter). The gradient can be computed by a classical back propagation: the differentiation chain rule is applied through the network, until the input word embeddeds.

4 Experiments

We conducted two sets of experiments. The goal of the first one is to see how well we can go by using the proposed architecture for Chinese SRL and its pre-processing tasks. The second experiment compared the decoding speeds between our system and the CRFs [18] based system. The four tasks (i.e. word segmentation, POS tagging, NER, and SRL) are evaluated by computing the standard F1-score, which is the harmonic mean of precision and recall.

4.1 Data Sets

We used different data sets to train and evaluate the performance of our neural networks on the word segmentation, POS tagging, and named entity recognition. For each task, we constructed a large data set that comprises data collected from multiple sources available. For the joint word segmentation and POS tagging, the corpus contains 3.15 million sentences (17.29 million characters), and for the NER, it contains 3.15 million sentences (17.29 million characters). As to Chinese SRL tasks, we manually collected and annotated a set of training data that includes at least six themes, such as dating, booking, banking, meeting, weather information, and notification. Each theme can be further divided into several sub-themes. For example, the system can process the booking messages for train tickets, airline tickets, and hotel reservations. We took about ten percent of each data set as the test data that was removed from the training set, and was not seen to the models.

4.2 Pre-Training Character Embeddings

Previous work showed that the performance could be improved by using word or character embeddings learned from large scale unlabeled data in many NLP tasks both in English [12][19] and Chinese [13][20]. Unsupervised pre-training guides the learning towards basins of attraction of minima that support better generalization [21]. We leveraged large unlabeled corpus to learn character embeddings, and then used these improved embeddings to initialize the character lookup tables of the networks. A Chinese Wikipedia corpus containing about 667 MB data was used to train the character embeddings by Word2Vec tool [22]. The Word2Vec tool provides two models to train the embeddings, and the preliminary experiments showed that the Skip-gram performed better than the CBOW. In all the experiments, we used the character embeddings that were learned from large unlabeled texts using the Skip-gram to initialize the neural networks.

4.3 Results

The results of four tasks are reported in **Table 2**, where the goals of this research are listed in the second column. We implemented our neural network-based models and others in Java, and all the experiments were run on the machines equipped with the same configurations. The hyper-parameters of all models are tuned on the development sets. All the test results were obtained over 5 runs with different random initialization. As shown in Table 2, the proposed neural networks boost the performances of all the tasks by a fairly significant margin, particularly for Chinese SRL (10% in average against the goal) and NER (9% in average).

Table 3 compares the decoding speeds on the test data from the bakeoff-3, Chinese Treebank (CTB) [23] for our neural network-based system and for two typical CRFs-based word segmentation systems, and **Table 4** compares the tagging speeds for our LSTM-based system with a CRFs-based system devel-

▼Table 2. Comparison with the goals of the research

Task	Goal	Model
Word segmentation (F1)	~ 85	⩾ 90
POS tagging (F1)	~ 80	⩾ 88
Named entity recognition (F1)	~ 75	⩾ 84
SRL (F1)	~ 70	⩾ 80

POS: part-of-speech SRL: semantic role labeling

▼Table 3. Comparison of the computational costs on Chinese word segmentation

System	Parameters	Time(ms)
Tsai et al. [24]	3027k	602
Zhao et al. [25]	3711k	859
Neural network	459k	49

▼Table 4. Comparison of the computational costs on SRL

System	Parameters	Time(ms)
CRFs-based system	28k	~600
LSTM (with 20 cells)	6k	~500

CRF: conditional random field LSTM: long short-term memory

oped by us on the Chinese SRL task. The decoding speed is reported in the average running time for processing one Chinese sentence.

In Table 4, we only list the number of additional parameters required for SRL task, which does not count the parameters used for the pre-processing tasks and those for storing the character embeddings. The running times reported in Table 4, however, are all the time required for Chinese SRL, including the necessary pre-processing tasks. Regardless of the differences in implementation, the neural network-based systems clearly run considerably faster than those based on the CRFs model that also require much more memory than our neural networks.

5 Conclusions

We have described a deep neural network-based model for the sequence labeling task in Chinese natural language processing, a variant of convolutional neural network architecture with dynamic k-max pooling layer for the character-level sentence classification, and a novel long short-term memory-based model for the Chinese SRL. The results of experiments show that the proposed neural networks performed reasonably well on various NLP tasks, highlighting the potential of the proposed networks for practical SRL and other similar tasks. The speed advantage of those neural networks makes them more competitive for the large-scale applications or applications requiring real-time response, especially for the applications deployed on smart phones.

References

[1] N. W. Xue, "Labeling Chinese predicates with semantic roles," *Computational Linguistics*, vol. 34, no. 2, pp. 225–255, 2008. doi: 10.1162/coli.2008.34.2.225.
[2] M. Surdeanu, S. Harabagiu, J. Williams, and P. Aarseth, "Using predicate-argument structures for information extraction," in *Proc. The Annual Meeting of the Association for Computational Linguistics*, Sapporo, Japan, Jul. 2003, pp. 8–15. doi: 10.3115/1075096.1075098.
[3] G. Melli, Y. Wang, Y. Liu, et al., "Descrition of SQUASH, the SFU question and summery handler for the DUC-2005 summarization task," in *Document Understanding Conference*, 2005.
[4] H. C. Boas, "Bilingual framenet dictionaries for machine translation," in *Proc. International Conference on Language Resources and Evaluation*, Jan. 2002, pp. 1364–1371.
[5] S. Narayanan, and S. Harabagiu, "Question answering based on semantic structures," in *Proc. 20th International Conference on Computational Linguistics*, Geneva, Switzerland, Aug. 2004. doi: 10.3115/1220355.1220455.
[6] S. Pradhan, W. Ward, K. Hacioglu, and J. H. Martin, "Semantic role labeling using different syntactic views," *in Proc. International Conference on Computational Linguistics*, USA, Michigan, Jun. 2005, pp. 581–588. doi: 10.3115/1219840.1219912.
[7] J. L. Packard, *The Morphology of Chinese: a Linguistic and Cognitive Approach*, Cambridge, United Kingdom: Cambridge University Press, 2004.
[8] N. Xue and M. Palmer, "Annotating the propositions in the Penn Chinese Treebank," in *Proc. The 2nd SIGHAN Workshop on Chinese Language Processing*, Sapporo, Japan, Jul. 2003, pp. 47–54. doi: 10.3115/1119250.1119257.
[9] Y. Bengio, "Learning deep architectures for AI," *Foundations and Trends in Machine Learning*, vol. 2, no. 1, pp. 1–127, 2009. doi: 10.1561/2200000006.
[10] G. F. Luger, *Artificial Intelligence: Structures and Strategies for Complex Problem Solving* 5th edition. New Jersey, USA: Addison Wesley, 2004.
[11] H. T. Ng, and J. K. Lou, "Chinese part-of-speech tagging: one-at-a-time or all-at-once? word-based or character-based?" in *Proc. Conference on Empirical Methods in Natural Language Processing*, Jul. 2004, pp. 277–284.
[12] R. Collobert, J. Weston, L. Bottou, et al., "Natural language processing (almost) from scratch," *Journal of Machine Learning Research*, vol. 12, no. 1, pp. 2493–2537, 2011.
[13] X. Zheng, H. Chen, and T. Xu, "Deep learning for Chinese word segmentation and POS tagging," in *Proc. Conference on Empirical Methods in Natural Language Processing*, Oct. 2013, pp. 647–657.
[14] N. Kalchbrenner, E. Grefenstette, and P. Blunsom, "A convolutional neural network for modelling sentences," in *Proc. Annual Meeting of the Association for Computational Linguistics*, Apr. 2014, pp. 655–665.
[15] S. Hochreiter, and J. Schmidhuber, "Long short-term memory," *Neural Computation*, vol. 9, no. 8, pp. 1735–1780, 1997. doi: 10.1162/neco.1997.9.8.1735.
[16] Z. Huang, W. Xu, and K. Yu. (2015, Aug.). Bidirectional LSTM-CRF models for sequence tagging [Online]. Available: http://arxiv.org/abs/1508.01991v1
[17] A. Graves, A. Mohamed, and G. Hinton. (2013, Mar.). Speech recognition with deep recurrent neural networks [Online]. Available: http://arxiv.org/abs/1303.5778
[18] M. Collins, "Discriminative training methods for hidden Markov models: theory and experiments with perceptron algorithms," in *Proc. Conference on Empirical Methods in Natural Language Processing*, 2002, pp. 1–8. doi:10.3115/1118693.1118694.
[19] R. Socher, C. C-Y. Lin, A. Y. Ng, and D. Christopher, "Parsing natural scenes and natural language with recursive neural networks," in *Proc. International Conference on Machine Learning*, Washington, USA, Jun. 2011, pp. 129–136.
[20] W. Pei, T. Ge, and B. Chang, "Max-margin tensor neural network for Chinese word segmentation," in *Proc. The Annual Meeting of the Association for Computational Linguistics*, 2014, doi:10.3115/v1/P14-1028.
[21] D. Erhan, Y. Bengio, A. Courville, P. Manzagol, et al., "Why does unsupervised pre-training help deep learning," *Journal of Machine Learning Research*, vol. 11, pp. 625–660, 2010.
[22] T. Mikolov, K. Chen, G. Corrado, and J. Dean. (2013, Jan.). "Efficient estimation of word representations in vector spaces [Online]. Available: http://arxiv.org/abs/1301.3781
[23] G. A. Levow, "The third international Chinese language processing bakeoff: word segmentation and named entity recognition," in *Proc. SIGHAN Workshop on Chinese Language Processing*, 2006, pp. 108–117.
[24] R. T. Tsai, H. C. Hung, C. Sung, H. Dai, et al., "On closed task of Chinese word segmentation: an improved CRF model coupled with character clustering and automatically generated template matching," in *Proc. The Fifth SIGHAN Workshop on Chinese Language Processing*, 2006, pp. 108–117.
[25] H. Zhao, C. N. Huang, and M. Li, "An improved Chinese word segmentation system with conditional random field," in *Proc. The Fifth SIGHAN Workshop on Chinese Language Processing*, 2006, pp. 162–165.

Manuscript received: 2016-11-24

Biographies

ZHENG Xiaoqing (zhengxq@fudan.edu.cn) received the Ph.D. degree in computer science from Zhejiang University, China. After then, he joined the faculty of School of Computer Science at Fudan University, China. He did research on semantic technology during his stay at the Information Technology Group, Massachusetts Institute of Technology (MIT), USA as an international faculty fellow from 2010 to 2011. His current research interests include natural language processing, deep learning, data integration and semantic web.

CHEN Jun (chen.jun_sh@zte.com.cn) received the Ph.D. Degree in signal and information processing from Xidian University, China. Since 2003, he joined ZTE corporation. His current research interests include image processing and deep learning technologies.

SHANG Guoqiang (shangguoqiang@zte.com.cn) received the B.S. Degree from Zhejiang University, China. Now, in ZTE his research interests include natural language processing, picture processing and so on.