ATLANTIS PRESS

*Advances in Intelligent Systems Research (AISR), volume 155*

2018 International Conference on Advanced Control, Automation and Artificial Intelligence (ACAAI 2018)

# Chinese Named Entity Recognition With Inception Architecture and Weight Loss

Jun zhao and Wei fang

School of Computer Science, Beijing University of Posts and Telecommunications, Beijing, P.R. China

***Abstract*—Named Entity Recognition(NER) is a very important part of many Natural Language Processing(NLP) tasks, but the accuracy rate of NER has not reached our expectation, especially in Chinese. Therefore, we studied NER task in Chinese social media. Compared to the previous papers on this dataset, we propose a new network structure that greatly improve the recognition performance. At the same time, we noticed that the recall rate of their experimental results is much lower than the precision rate, so we also explored a method to mitigate this situation by changing the loss function. Finally, the experimental results of ours with the new loss function obtained not only higher recall rates, but also significant speedup in training phase compared to the state-of-the-art methods.**

***Keywords-Named entity recognition(NER); Natural Language Processing (NLP); Chinese social media;Deep Learning;***

## I. INTRODUCTION

Named entity recognition(NER) has always been an indispensable step in the information extraction task. So, people also have done a lot of research in this area. The methods of NER is developed from the traditional feature-based like conditional random fields(CRF) and lexicographic methods to the combining deep learning with CRF. The accuracy of recognition is also increasing. Like other natural language processing(NLP) tasks, in the domain of NER, the neural network was first introduced by [1], After that researcher are constantly exploring different neural network structures, from the shallow neural network to the long short-term memory(LSTM) [2][3], bidirectional long short-term memory(BLSTM) [4], and then the attention mechanism was used with the LSTM [5]. Now, the state-of-the-art performance generally implemented by CNN+LSTM+CRF [8]. However, because of that there are great differences in different NER fields and linguistic backgrounds, people must explore different methods to handle these differences. One of the most flexible ways is to change the network input. The most common input includes word embedding and character embedding.

In the field of NER, English tasks is the most attractive direction, and the achievement of English is also the greatest. Compared to English tasks, Chinese NER is often more challenging. The main reasons are the following: a) There is no delimiter between Chinese words. b) Chinese words are made up of a single word (Chinese character) and the combination of characters is numerous. So, the Chinese words have a large dictionary. Not the same as English NER task that label the words directly, the state-of-the-art Chinese model is usually labelling the character. But there is a problem in embedding a character directly, that's the same character, they may be unrelated in meaning, have the same embedding. That is not what we expected. Peng and Dredze [6] explored several types of representations for Chinese, including pre-segmenting the input to obtain words, using character embeddings, and a combined approach that learned embeddings for characters based on their position in the word. This final representation yielded the largest improvements. On this basis, [7] also tried to combine the word segmentation system with the NER task, and further improved the performance. In the aspect of network structure, [8] use Convolutional Neural Networks(CNN) to get the result of character embedding, and then join the CNN outputs with word-embedding as the input to LSTM-CRF layer and obtain state-of-the-art performance. However, compare with English, Chinese NER still has a lot of room for promotion.

In this paper, we explored a new network structure and embedding approach. In terms of network structure, the inspiration is mainly from the inception network [9][10]. Although some researchers have already introduced CNN ([11]) into NER tasks [8], they are all single size convolution kernels. In the field of computer vision, researcher attempted to capture different levels of features by introducing different size convolution kernels. Therefore, we introduced different size of convolution kernels to capture n-gram semantic information of different window size. And the results show that the method is effective. In the aspect of word-embedding, When Peng and Dredze [6] are embedded in words, the same character in different words has a completely different embedding. Then, to alleviate the complete irrelevance of the same character, they introduced language model. We tried another different embedding approach without additional language models, and achieved the same performance.

Finally, in the training process, we noticed that the final recall is much lower than the precision and the accuracy was increase very slowly. Therefore, we modified the loss function, the training process has been accelerated and the recall has increased.

## II. MODEL

In this section, we will describe the neural network structure we use in this paper. It will be introduced in turn from the top to the output

### *A. CNN Layer*

CNN is widely used in many NLP tasks, such as text categorization, some seq2seq tasks, NER and so on. A reasonable explanation is that CNN has the role of n-gram in some degree. To get n-gram with different window size, we first used multi dimension convolution layer in NER task. The basic

convolution unit is the same as Inception [9]. It is a parallel convolution kernel (also known as Bottleneck Layer), which is proposed by GoogleNet. Figure I is the structure that we used.

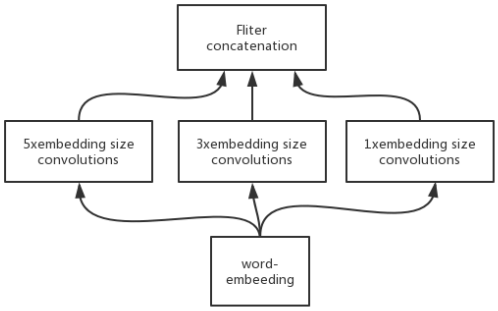


FIGURE I. THE BASIC UNIT STRCTURE THAT WE USED, IT CONTAINS THREE SIZES OF CONVOLUTION KERNELS

It can get the semantic information of different window size and then link them into the next layer of neural networks. The experimental results show that the inception structure improved the performance of NER.

### B. *BSTM Layer*

RNN [12] is a simple extension of the neural network, and has made a lot of amazing achievements over the past few years. LSTM [13] is an improved model of RNN. LSTMs are variants of RNNs designed to cope with these gradient vanishing problems. Basically, a LSTM unit is composed of three multiplicative gates which control the proportions of information to forget and to pass on to the next time step. In this paper, we use bidirectional LSTM, for many sequence labeling tasks it is beneficial to have access to both past (left) and future (right) contexts, as the basic unit which can greatly improve the performance of sequence annotation. We use the following implementation:

$$
\begin{aligned}
i_t &= \sigma(w_{xi}x_t + w_{hi}h_{t-1}) + b_i \\
c_t &= (1 - i_t)\otimes c_{t-1} + \\
&\quad i_t \otimes \tanh(w_{xc}x_t + w_{hc}h_{t-1}) + b_c \\
o_t &= \sigma(w_{xo}x_t + w_{ho}h_{t-1}) + b_o \\
h_t &= o_t \otimes \tanh(c_t)
\end{aligned}
$$

where σ is the element-wise sigmoid function, and ⊗ is the element-wise product.

### C. *CRF Layer*

Like LSTM, CRF is also a very popular method for sequence annotation. It is beneficial to consider the correlations between labels in neighborhoods and jointly decode the best chain of labels for a given input sentence. Therefore, we model label sequence jointly using a conditional random field (CRF) [14], instead of decoding each label independently.

Each of the maximum groups in the CRF contains two nodes, and the current node is $i$, the previous node is $i-1$. The characteristic granularity is the granularity of the largest group, and it is the two adjacent nodes $\{i, i-1\}$. Generally, the characteristics are divided into two categories, on the side and on the node. The m dimension features on the edge are expressed in $t_m$, and the l features on the node are expressed in $s_l$. The feature vector of each group is $(t_1, t_2, \ldots, t_M, s_1, s_2, \ldots, s_L)$. For convenience, it is usually being represented as $(f_1, f_2, \ldots, f_M, f_{M+1}, f_{M+2}, \ldots, f_{M+L})$. Detailed formulas are as follows:

$$
P(y|x) = \frac{1}{Z(x)} exp \sum_{k=1}^{K} w_k f_k(y, x)
$$

$$
Z(x) = \sum_{y} exp \sum_{k=1}^{K} w_k f_k(y, x)
$$

$$
f_k(y, x) = \sum_{k=1}^{K} f_k(y_{i-1}, y_i, x, i)
$$

$P(y|x)$ represented the probabilistic model for sequence CRF, $f_k(y, x)$ represented the feature function. For CRF training, we use the maximum conditional likelihood estimation.

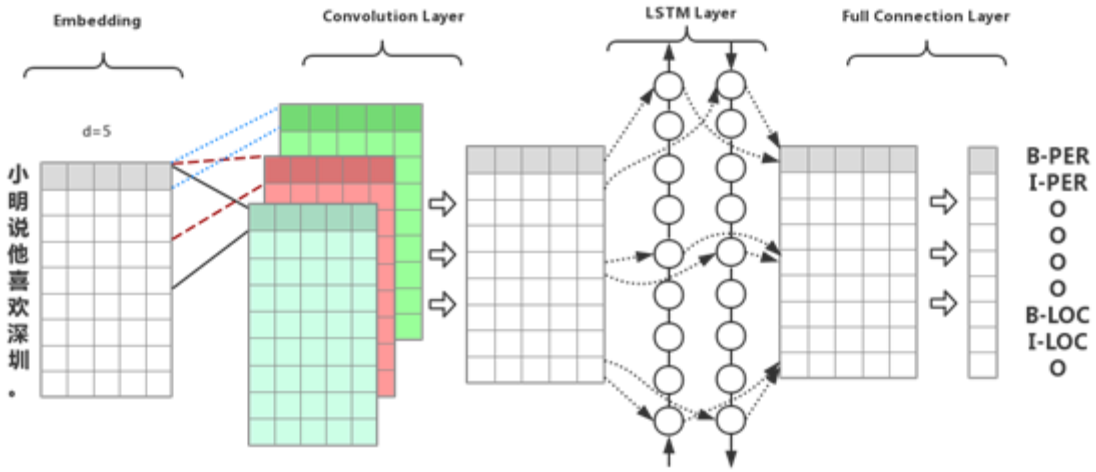


FIGURE II: THE STRUCTURE OF NEURAL NETWORK. THE INPUT VECTOR IS CHARACTER EMBEDDING PLUS POSITION EMBEDDING. WHEN THE INPUT VECTOR IS CALCULATED BY THE CNN LAYER, THE RESULTS ARE INPUT TO THE BLSTM STRUCTURE

### D. Input Embedding

We combine character embedding and position embedding, and set the same dimensions. After obtaining the character embedding and position embedding, add their two values to get the final representation of the word.

- character embedding

For character embedding, we first define a fixed length vector for each word first, and then use the back-propagation algorithm to iterate the update vector value in the later training process.

- position embedding

It's not the same as [6], We separate the position information from the character embedding [22] [23]. The positional encodings have the same dimension as the embeddings, so that the two can be

summed. There are many choices of positional encodings, learned and fixed.

In this work, we use sine and cosine functions of different frequencies:

$f(x_i,w)$ denote the normalized probability function and it j dimension value is $f(x_i,w)^j$ . $\theta(y_i)$ denote the weight function.Generally, we taken a sampling method like under sampling or oversampling [15] for this kind of problem. But, in NER tasks, the smallest element of the data is sentence. We cannot add a single label without affecting the others. So, when we calculating the loss in this paper, for different categories, we according to their distribution in the data set to set the corresponding weight. And this modification can mitigate the impact of uneven distribution of the labels.

$$PE_{(pos,2i)} = \sin(pos/10000^{2i/d_{model}})$$

$$PE_{(pos,2i+1)} = \cos(pos/10000^{2i/d_{model}})$$

where pos is the position and i is the dimension. That is, each dimension of the positional encoding corresponds to a sinusoid. The wavelengths form a geometric progression from $2\pi$ to $10000 \cdot 2\pi$. We chose this function because we hypothesized it would allow the model to easily learn to attend by relative positions, since for any fixed offset k, $PE_{pos+k}$ can be represented as a linear function of $PE_{pos}$.

Figure II show the final model structure.

### E. Loss Function

As mentioned above, we found that the gap between the recall and precision is very large and the speed of training is very slow, so we adjust the loss function. The new loss function is shown as follows:

$$L(x,y,w) = -\sum_{i=1}^{N} \theta(y_i) \prod_{j=1}^{K} y_i^j \log \cdot f(x_i,w)^j$$

$$\theta(y_i) = \gamma \cdot I(y_i)$$

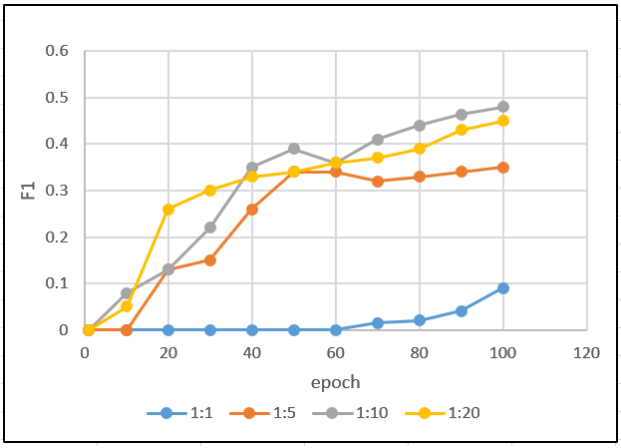


FIGURE III: RESULTS OF ADJUST DIFFERENT WEIGHTS, '1:1' DENOTE THE RATIO OF OTHER TAG TO NER TAG

The results show that this change greatly improve the training speed,

TABLE I. SHOW THE RESULT. NER RESULTS FOR VALIDATION SET AND TEST SET

| | | Dev | | | Test | | |
|---|---|---|---|---|---|---|---|
| | **Method** | **Procision** | **Recall** | **F1** | **Procision** | **Recall** | **F1** |
| 1 | LSTM | 47.05 | 36.36 | 41.02 | 39.06 | 42.37 | 36.23 |
| 2 | LSTM-CRF | 46.06 | 45.78 | 45.92 | 45.36 | 35.47 | 39.81 |
| 3 | CNN-LSTM | 60.27 | **50.87** | **55.17** | 47.52 | **51.15** | **49.26** |
| 4 | Joint Train LSTM +Emb | **65.32** | 40.78 | 50.21 | **67.32** | 35.45 | 46.44 |

## III. EXPERIMENTS

In this paper, we calculated f1 values for name mentions and nominal mentions respectively. The results shown in Figure 1. And we also compared the effect of different weights on the final experiment results and the experimental convergence rate.

### A. Data sets

We use the same training, development and test splits as [6] for word segmentation and [7] for NER。corpus includes 1,890 messages sampled from Weibo between November 2013 and December 2014.

### B. *Effect of Loss Function*

In experiment, we experimented multiple sets of weights, and the results show that the ratio of the weight between NER with other tag is 10:1, the highest F1 has been obtained, and the convergence speed of experiment also increases by 4-8 times.

We have done comparative experiments on BLSTM, BLSTM-CRF and CNN-BLSTM respectively, and the results show that, all of them are very sensitive to weight. During the experiment, a problem that the final accuracy rate did not converge when the weight ratio was not suitable has arisen in some times.

All the detail show in Figure 2.

### C. *Result*

Table I shows the performance of different network structures on training set, test set and Validation set. We found that there is a serious over-fitting in the training process. Later, we add the L2 regularization [16] [17] and batch normalization [18] [19] techniques to the CNN [20] [21], but the over-fitting phenomenon still exists. The main reason is that the training set is a small scale. Thus, by comparing the simulation results we finally adopt position embedding. The simulation shows that we had about 3%-5% of the upgrade.

We explore the effect of convolution kernels [24] of different sizes on the final experiment. Through experiments we find that CNN-BLSTM obtained the best performance on the test set validation set.

## IV. CONCLUSION

In this paper, we have tried to replace the previous irrelevant position embedding with new position embedding technology and get state-of-the-art performance. At the same time, we capture the semantic information of different window size by using inception multi-size convolution kernel to enhance the recognition performance of NER. In the end, we obtain a higher recall through the weight loss to increase the F1 value and speed up the convergence process.

There is still room for improvement. Mainly in the value of the weight, here we choose a common method like the selection of hyperparameters in neural networks, such as random selection and grid search. But this is a clumsy way. Subsequent research can translate these hyper paragraphs into network variables and then use backpropagation to automatically update the parameters, a similar work in the selection of neural network hyper paragraph has been recently tried.

## ACKNOWLEDGMENT

I'm grateful to professor Fang Wei for some constructive suggestions put forward in the experiment, which play an important role in improving the experimental results. And, we would like to thank [6] for releasing source code, some of the technical details that are not mentioned in their paper could find the answer on the code, it's very important.

## REFERENCES

[1] Collobert R, Weston J, Karlen M, et al. Natural Language Processing (Almost) from Scratch[J]. Journal of Machine Learning Research, 2011, 12(1):2493-2537.

[2] Hochreiter S, Schmidhuber J. Long Short-Term Memory[J]. Neural Computation, 1997, 9(8):1735.

[3] Dyer C, Ballesteros M, Wang L, et al. Transition-Based Dependency Parsing with Stack Long Short-Term Memory[J]. Computer Science, 2015, 37(2):321–332.

[4] Lample G, Ballesteros M, Subramanian S, et al. Neural Architectures for Named Entity Recognition[J]. 2016:260-270.

[5] Luong M T, Pham H, Manning C D. Effective Approaches to Attention-based Neural Machine Translation[J]. Computer Science, 2015.

[6] Peng N, Dredze M. Named Entity Recognition for Chinese Social Media with Jointly Trained Embeddings[C]// Conference on Empirical Methods in Natural Language Processing. 2015:548-554.

[7] Peng N, Dredze M. Improving Named Entity Recognition for Chinese Social Media with Word Segmentation Representation Learning[C]// Meeting of the Association for Computational Linguistics. 2016:149-155.

[8] Ma X, Hovy E. End-to-end Sequence Labeling via Bi-directional LSTM-CNNs-CRF[J]. 2016.

[9] Szegedy C, Liu W, Jia Y, et al. Going deeper with convolutions[C]// Computer Vision and Pattern Recognition. IEEE, 2015:1-9.

[10] Szegedy C, Ioffe S, Vanhoucke V, et al. Inception-v4, Inception-ResNet and the Impact of Residual Connections on Learning[J]. 2016.

[11] Lecun Y, Bengio Y. Convolutional networks for images, speech, and time series[M]// The handbook of brain theory and neural networks. MIT Press, 1998.

[12] Mikolov T, Karafiát M, Burget L, et al. Recurrent neural network based language model[C]// INTERSPEECH 2010, Conference of the International Speech Communication Association, Makuhari, Chiba, Japan, September. DBLP, 2010:1045-1048.

[13] Hochreiter S, Schmidhuber J. Long Short-Term Memory[J]. Neural Computation, 1997, 9(8):1735.

[14] Lafferty, John D, McCallum, et al. Conditional Random Fields: Probabilistic Models for Segmenting and Labeling Sequence Data[M]// Departmental Papers (CIS). 2001.

[15] Chawla N V, Bowyer K W, Hall L O, et al. SMOTE: synthetic minority over-sampling technique[J]. Journal of Artificial Intelligence Research, 2011, 16(1):321-357.

[16] Ng A Y. Feature selection, L1 vs. L2 regularization, and rotational invariance[C]// ACM, 2004:78.

[17] Schölkopf B, Platt J, Hofmann T. Efficient Structure Learning of Markov Networks using L1-Regularization[C]// Advances in Neural Information Processing Systems 19, Proceedings of the Twentieth Conference on Neural Information Processing Systems, Vancouver, British Columbia, Canada, December. DBLP, 2006:817-824.

[18] Cooijmans T, Ballas N, Laurent C, et al. Recurrent Batch Normalization[J]. 2016.

[19] Li Y, Wang N, Shi J, et al. Revisiting Batch Normalization For Practical Domain Adaptation[J]. 2016.

[20] Zhao Z Q, Bian H, Hu D, et al. Pedestrian Detection Based on Fast R-CNN and Batch Normalization[J]. 2017:735-746.

[21] Simon M, Rodner E, Denzler J. ImageNet pre-trained models with batch normalization[J]. 2016.

[22] Vaswani A, Shazeer N, Parmar N, et al. Attention Is All You Need[J]. 2017.

[23] Gehring J, Auli M, Grangier D, et al. Convolutional Sequence to Sequence Learning[J]. 2017.

[24] Dai J, Qi H, Xiong Y, et al. Deformable Convolutional Networks[J]. 2017