Chinese NER with Step-wise Attention

Xinyue Chen and Fan Xie
Department of Computer Science and Engineering
Shanghai Jiaotong University
Minhang, Shanghai, 200240
kiwisher@sjtu.edu.cn, fanxie806@gmail.com

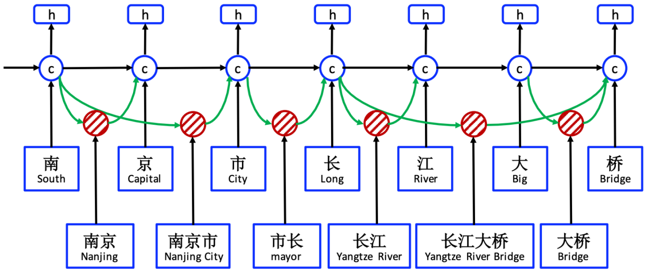

Figure 1: An example for lattice model.

Abstract

We investigate a new model for Chinese NER task. Based on the previous state-of-the-art model lattice model for this task, we build a model using attention mechanism that better leverages word-level and character-level information.

Our model outperforms the lattice model on two datasets without fine tuning of hyper parameters, suggesting its robustness in various corpus.

We borrow much from the lattice model: we also encode a sequence of input characters as well as all potential words that match a lexicon in a sentence. But we further use our step-wise attention instead of the lattice structure to weigh relevance scores of candidate characters and words from a sentence for a more expressive model. We claim that our model helps reduce perplexity and bring a more succinct understanding of corpus. And the results in several corpus proved this assumption.

1 Introduction

As a fundamental task in information extraction, named entity recognition (NER) has received constant research attention over the recent years. The task has traditionally been solved as a sequence labeling problem, where entity boundary and category labels are jointly predicted. The current stateof-the-art for English NER has been achieved by using LSTM-CRF models(Lample et al. 2016; Ma and Hovy 2016; Chiu and Nichols 2015; Liu et al. 2018) with character information being integrated into word representations.

Chinese NER is correlated with word segmentation. In particular, named entity boundaries are also word boundaries. One intuitive way of performing Chinese NER is to perform word segmentation first, before applying word sequence labeling. The segmentation → NER pipeline, however, can suffer the potential issue of error propagation, since NEs are an important source of OOV in segmentation, and incorrectly segmented entity boundaries lead to NER errors. This problem can be severe in the open domain since crossdomain word segmentation remains an unsolved problem (Liu et al. 2018; Jiang et al. 2013; Huang, Sun, and Wang 2017). It has been shown that character-based methods outperform word-based methods for Chinese NER.

One drawback of character-based NER, however, is that explicit word and word sequence information is not fully exploited, which can be potentially useful. To address this issue, we integrate latent word information into character-based LSTM-CRF by representing lexicon words from the sentence using step-wise attention mechanism, which is illustrated in this graph. We align two levels of inputs, word-level and character-level, by matching a sentence with a large automatically-obtained lexicon. As a result, word sequences such as “长江大桥 (Yangtze River Bridge)”，“长江 (Yangtze River)” and “大桥 (Bridge)” can be used to disambiguate potential relevant named entities in a context, such as the person name “江大桥 (Daqiao Jiang)”.

Since there are an exponential number of ways for word segmentation, we propose a step-wise attention mechanism for automatically controlling information flow from the beginning of the sentence to the end. As shown in Section 4, step-wise attention is used to dynamically route information from different paths to each character. Trained over NER data, LSTM with step-wise attention can learn to find more useful words from context automatically for better NER performance. This model has the advantage of leveraging explicit word information over character sequence labeling without suffering from segmentation error.

We change from lattice to step-wise attention fol-

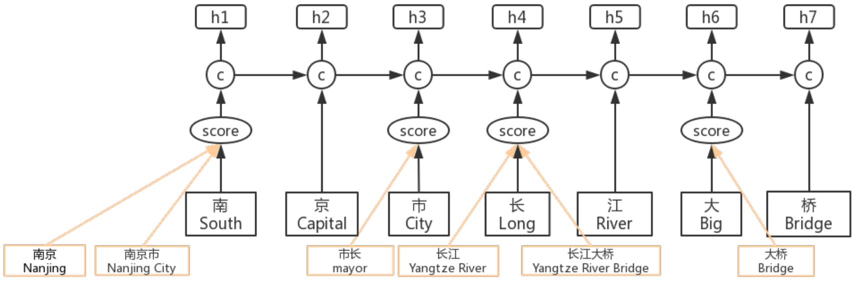

Figure 2: An example for our model.

lowing the pattern that human read sentences– humans generally make assumptions for the next incoming words and their understandings of the context is constantly changing throughout the whole reading process.

Results show that our model outperforms the previous state-of-the-art lattice model, giving the best results over two mainstream Chinese NER datasets.

2 Related Work

Our work is in line with existing methods using neural network for NER. (Hammerton 2003) attempted to solve the problem using a unidirectional LSTM, which was among the first neural models for NER. Most recent work leverages an LSTM-CRF architecture. (Huang, Xu, and Yu 2015) uses hand-crafted spelling features; (?) and (Chiu and Nichols 2015) use a character CNN to represent spelling characteristics; (Lample et al. 2016) use a character LSTM instead.

Character sequence labeling has been the dominant approach for Chinese NER (Dong et al. 2016). There have been explicit discussions comparing statistical word-based and character-based methods for the task, showing that the latter is empirically a superior choice (Liu, Zhu, and Zhao 2010; Li et al. 2014). With proper representation settings, the same conclusion holds for neural NER.

How to better leverage word information for Chinese NER has received continued research attention, where segmentation information has been used as soft features for NER (Peng and Dredze 2015; He and Sun 2017), and joint segmentation and NER has been investigated using dual decomposition(Xu et al. 2014), multi-task learning(Peng and Dredze 2016). However, the above methods can be affected by segmented training data and segmentation errors.

External sources of information has been leveraged for NER. In particular, lexicon features have been widely used (Collobert et al. 2011; Huang, Xu, and Yu 2015; Luo et al. 2015). (Rei 2017) uses a word-level language modeling objective to augment NER training, performing multi-task learning over large raw text. (Peters et al. 2017) pretrain a character language model to enhance word representations. (Yang, Zhang, and Dong 2017) exploit cross-domain and cross-lingual knowledgevia multi-task learning. We leverage external data by pretraining word embedding lexicon over large automatically-segmented texts, while semi-supervised techniques such as language modeling are orthogonal to and can also be used for our model.

More recently, (Zhang and Yang 2018) came with a lattice model, aiming at automatically controlling information flow between word-level and character-level embeddings. However, this model suffers from complexity and confusion entailed by the lattice structure itself. Also, lattice model's fusion of word-level and character-level information is problematic in that the information of a word does not flow into the mainstream (the character-based bi-LSTM) until the timestamp of the last character in the word.

Based on these observations, we design our model with step-wise attention. Furthermore, we add sentence representation as a global guide in the attention mechanism.

3 Proposed Model

There are two primary deficiencies in the lattice model.

- **complexity**

  As shown in this figure, the red and blue circles are all LSTM cells, which are costly and time-consuming. Moreover, we claim that this task can be expressed and explained in a more succinct model.

- **fusion incoherence**

  In the lattice model, the information of a word does not flow into the mainstream (the character-based bi-LSTM) until the timestamp of the last character in the word. For example, in this figure, the information of ” 长江大桥” (Yangtze River Bridge) flows into the character-level forward LSTM in the last timestamp, bring no direct impact on the labeling of the previous timestamps(” 长”, ” 江”, ” 大”).

  It is true that correctly defining the label of ” 桥” helps with the labeling of the previous characters since the labeling of neighboring characters are interrelated. But if the structure becomes more complex, and word-edges become dense in a sentence. Confusion arises inevitably.

Therefore, we propose the following model to solve the problem.

3.1 Step-wise Attention

The overall structure of our model with step-wise attention is shown in this figure, which can be viewed as an extension of the character based model.

Shown in this figure, the input to the model is a character sequence $c_1, c_2, ..., c_m$, together with all character subsequences that match words in a lesion D. Using $w^d_{b,e}$ to denote such a subsequence that begins with character index b and ends with character index e, the segment $w^d_{1,2}$ in this figure is "南京 (Nanjing)" and $w^d_{7,8}$ is "大桥 (Bridge)" . **Note that from now on, we use $x^w_{b,e}$ to denote only these words that not only exists in the sentence as a subsequence, but also is a valid word in our external lexicon.**

This revision of input words is in line with our assumption of the lattice model. We find it more sensible to feed all possible words at the timestamp of its starting character. **This fusion method is analogous to human reading a paragraph**– people always make assumptions in the process of reading. And in this reading process, people tend to be more accurate at predicting the next words and gain more understandings of the corpus. So, the model will predict what is the most likely way to segment the part that it is now processing based on the previous hidden state. The prediction is used to score the candidates words (representing possible word segmentation in some sense).

Four types of vectors are involved in the model, namely input vectors, output hidden vectors, cell vectors and gate vectors. As basic components, a character input vector is used to represent each character c_j:

$$x^c_j = e^c(c_j) \tag{1}$$

The basic recurrent structure of the model is constructed using a character cell vector and a hidden vector h^c_j on each c_j, where c^c_j serves to record recurrent information flow from c_j to the end of the sentence and h^c_j is used for CRF sequence labelling.

The computation of c^c_j now considers lexicon subsequences $w^d_{b,e}$ in the sentence. In particular, each subsequence $w^d_{b,e}$ is represented using $x^w_{b,e} = e^w(w^d_{b,e})$ where e^w denotes the word embedding lookup table. If in a certain sentence, we have n possible words in our lexicon that match with a part of this sentence and start with c_b, The set of $x^w_{b,e}$ as well as x^c_b will be concatenated together to form a new input $x'^{w}_{b,e}$. To simplify the notation, we denote x^c_b as $x^w_{b,b}$.

$$x^w_b = [x^w_{b,e_1}, x^w_{b,e_2}, .., x^w_{b,e_n}, x^w_{b,e_{n+1}}] \tag{2}$$

where h_{b-1} denotes the hidden state of the timestamp just before b, and $x^w_{b,e_{n+1}}$ is $x^w_{b,b}$, i.e. the character embedding. For example, the input of the timestamp of " 长" is the concatenation of the embeddings of " 长", "长江" and " 长江大桥".

Then attention is used to compute the score of each $x^w_{b,e}$:

$$score_{b,e} = h_{b-1} W x^w_{b,e} \tag{3}$$

After scoring the $x'^{w}_{b,e}$, $x'^{w}_{b,e}$ becomes:

$$x'^{w}_{b} = \sum_i score_{b,e_i} x'^{w}_{b,e_i} \tag{4}$$

In addition, a word cell $c^w_{b,e}$ is used to represent the recurrent state of $x'^{w}_{b,e}$ from the current character to the end of the sentence. The value of $c^w_{b,e}$ and h^c_b is calculated by:

$$\begin{bmatrix} i^w_b \\ o^w_b \\ \tilde{c}^w_b \end{bmatrix} = \begin{bmatrix} \sigma \\ \sigma \\ \tanh \end{bmatrix} \left(\mathbf{W}^{w\mathrm{T}} \begin{bmatrix} x'^{\,w}_b \\ h^c_b \end{bmatrix} + \mathbf{b}^w \right) \tag{5}$$

where $i^w_{b,e}$ and $o^w_{b,e}$ are a set of input and output gates. In our implementation, the forget gate is simply given as:

$$f^w_b = 1 - i^w_b \tag{6}$$

We find that this technique helps with the convergence of LSTM models.

3.2 Sentence-level Supervision

As h_{b-1} only contains the information in the previous timestamps (in the perspective of forward LSTM, backward LSTM is similar), we introduce sentence embedding as a global guide to supervise the scoring process in our step-wise attention mechanism.

The intuition is that, even if the information from the previous timestamps might be insufficient, especially in the very first timestamps in a long sentence, with the aid of sentence information, the model will be more capable of comprehending the context and thus predicting the most possible segmentation.

Another thing to consider is how to integrate h_{b-t} with the sentence embedding. We need to integrate these two channels efficiently, without staining their individual semantic information.

We leave a majority of this part to Section 4 for detailed discussions.

4 Discussions

4.1 Sentence Embedding Extraction

There are generally three methods in our consideration to extract sentence embeddings. We denote sentence embedding as x^s

- **Self attention**

  Self attention is a special case of multi-head attention (Vaswani et al. 2017) where

$$Q = K = V \tag{7}$$

Datasets	**Models**	**P**	**R**	**F1**
OntoNotes	Lattice	76.35	71.56	73.88
	Sentence-level supervision	**78.05**	**73.30**	**73.93**

Table 1: Main results on OntoNotes

Datasets	**Models**	**P**	**R**	**F1**
MSRA	Lattice	93.57	92.79	93.18
	Model 1	94.62	93.13	93.59
	Model 2 shared concat	**95.34**	**93.28**	93.59
	Model 2 shared mul	94.70	92.08	92.74
	Model 2 separate concat	95.03	93.01	93.51
	Model 2 separate mul	95.01	93.02	**93.67**
	Model 2 separate concat $+W_{w \rightarrow c}$	94.33	92.91	92.80

Table 2: Main results on MSRA

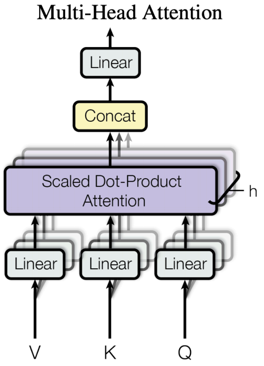

Figure 3: Multi-head attention.

Instead of producing a vector of length $len(e^c) * len(sentence)$, we modify it to produce a fixed-length vector to help it blend in with our model.

The sentence embedding is thus more expressive of the interrelations of characters inside a sentence.

- **The last hidden state from LSTM**

  This is a more popular way to encode a sentence, which is in line with the encoder-decoder model in NLP tasks. To produce a more expressive representation, we concatenate the last hidden state from both the forward and backward LSTM, so that the embedding will be less biased towards the beginning and end of a sentence.

- **The set of hidden states from LSTM**

  Instead of using only the last hidden state, we can also utilize all the hidden states from the LSTM.

  We can further score the hidden states from a naive LSTM imposing on the sentence, using the hidden state from our step-wise attention model (i.e. h_{b-1}) as a judge. Then the weighted average of the elements inside the set of hidden states is integrated with h_{b-1}.

4.2 Integration of h_{b-1} and x^s

There are generally three ways to integrate two sources of information.

- *element-wise add*
- *element-wise multiply*
- *concatenate*

We believe that *concat* and *mul* operations might be more informative than add, because it is too simple to prevent from detailed information losing.

Though *concat* operation may result in high dimensions, it could keep more information for downstream models to select from. The superiority of *mul* might be due to element-wise product being capable of modeling the interactions and eliminating distribution differences between h_{b-1} and x^s, which is intuitively similar to endowing sentence-aware attention over the hidden state.

So we try both *element-wise multiply* and *concatenate*.

4.3 Embedding Space Transformation

Intuitively, word embeddings and character embeddings are in two different spaces. So it is a natural idea to apply linear transformation before treating them equally in the step-wise attention model.

Therefore, we try to transform word embeddings with matrix multiplication in order to project them into the space of character embeddings.

It has been proved that orthogonal matrix helps with space transformation. So we try an orthogonal-initialized matrix, $W_{w\rightarrow c}$. But experiments show that this method hurt our model's performance. Possible reasons are:

- **The embedding vectors are fine-tuned during training.**

  Since embeddings are to be back-proped. The gap between different spaces might be bring to a semantic harmony.

- **Redundancy**

  Adding an additional parameter $W_{w\rightarrow c}$ may slow down the convergence of model.

The same as model mentioned before, scoring function is used in the model. The integrated hidden state scores the x'.

2 Experiments

We carry out an extensive set of experiments to investigate the effectiveness of our model in two named datasets for Chinese NER.

4.1 Experiment Settings

We borrow much of the settings from (Zhang and Yang 2018), but we achieve higher performance with our step-wise attention without tuning hyper parameters.

Dataset

We used two datasets in this experiment: OntoNotes 4 and MSRA.

Embeddings

Word embeddings is pretrained by word2vec over automatically segmented Chinese Giga-Word, obtaining 704.4k words in a final lexicon. In particular, the number of single-character, two- character and three-character words are 5.7k, 291.5k, 278.1k, respectively. Word embeddings are fine-tuned during NER training.

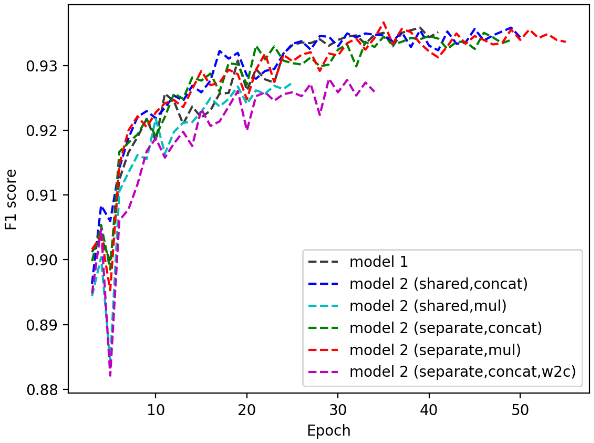

Figure 4: F1 against training iteration number on MSRA.

Character and character bigram embeddings are pretrained on Chinese Giga-Word using word2vec and finetuned at model training.

Result Notations

Here we clarify our notations used for reporting our experiment results.

- **Model 1**: Our step-wise attention model without sentence-level supervision
- **Model 2**: Our step-wise attention model with sentence-level supervision
- **shared / separate**: Whether a sentence embedding is shared among both directions of LSTM or for each LSTM use a separate sentence embedding.

  Due to time limit, we implement sentence embedding extraction via self attention.
- **w2c / $W_{w\rightarrow c}$**: Indicate whether a model uses word-to-character embedding space transformation.

4.2 Final Results

MSRA

The results of MSRA are shown in Table **x**. Among all the tested models, the values of precision and recall of step-wise attention model with concatenation are significantly higher than lattice model. Sentence-level supervision model with multiplication and without $W_{w\rightarrow c}$ has the highest F1 value. And model with $W_{w\rightarrow c}$ gives a F1-score of 92.38% doesn't have a better performance than the same basic model without $W_{w\rightarrow c}$. This fact tells us that the existence of embedding space transformation does not help.

This figure shows the F1-score of models on MSRA dataset against the number of training iterations. As shown in the figure, the value of model 1 with *mult* and no w2c and the value of model 2 with w2c and concatenation are generally smaller than the other models.

OntoNotes Table **x** shows the results on OntoNotes dataset. The P, R, and F1 values of sentence-level supervision model are higher than lattice model. Due to time limit, we only implement our model2 with separate sentence embedding on this dataset. Therefore, we can draw a conclusion that sequence-level supervision model outperforms lattice model.

Acknowledgements

We thank Professor Jiang for presenting us with this opportunity and guide us in digging the specific problem. Also, we would like to express our gratitude for the author of (Zhang and Yang 2018), Jie Yang, who kindly exchange ideas with us.

References

[Chiu and Nichols 2015] Chiu, J. P. C., and Nichols, E. 2015. Named entity recognition with bidirectional lstm-cnns. *CoRR* abs/1511.08308.

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

[Dong et al. 2016] Dong, C.; Zhang, J.; Zong, C.; Hattori, M.; and Di, H. 2016. Character-based lstm-crf with radical-level features for chinese named entity recognition. In *NLPCC/ICCPOL*.

[Hammerton 2003] Hammerton, J. 2003. Named entity recognition with long short-term memory. In *Proceedings of the Seventh Conference on Natural Language Learning at HLT-NAACL 2003*.

[He and Sun 2017] He, H., and Sun, X. 2017. A unified model for cross-domain and semi-supervised named entity recognition in chinese social media. In *AAAI*.

[Huang, Sun, and Wang 2017] Huang, S.; Sun, X.; and Wang, H. 2017. Addressing domain adaptation for chinese word segmentation with global recurrent structure. In *Proceedings of the Eighth International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, 184–193. Asian Federation of Natural Language Processing.

[Huang, Xu, and Yu 2015] Huang, Z.; Xu, W.; and Yu, K. 2015. Bidirectional lstm-crf models for sequence tagging. *CoRR* abs/1508.01991.

[Jiang et al. 2013] Jiang, W.; Sun, M.; Lü, Y.; Yang, Y.; and Liu, Q. 2013. Discriminative learning with natural annotations: Word segmentation as a case study. In *Proceedings of the 51st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, 761–769. Association for Computational Linguistics.

[Lample et al. 2016] Lample, G.; Ballesteros, M.; Subramanian, S.; Kawakami, K.; and Dyer, C. 2016. Neural architectures for named entity recognition. In *Proceedings of the 2016 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, 260–270. Association for Computational Linguistics.

[Li et al. 2014] Li, H.; Hagiwara, M.; Li, Q.; and Ji, H. 2014. Comparison of the impact of word segmentation on name tagging for chinese and japanese. In *LREC*.

[Liu et al. 2018] Liu, L.; Shang, J.; Ren, X.; Xu, F.; Gui, H.; Peng, J.; and Han, J. 2018. Empower sequence labeling with task-aware neural language model.

[Liu, Zhu, and Zhao 2010] Liu, Z.; Zhu, C.; and Zhao, T. 2010. Chinese named entity recognition with a sequence labeling approach: Based on characters, or based on words? In *ICIC*.

[Luo et al. 2015] Luo, G.; Huang, X.; Lin, C.-Y.; and Nie, Z. 2015. Joint entity recognition and disambiguation. In *EMNLP*.

[Ma and Hovy 2016] Ma, X., and Hovy, E. H. 2016. End-to-end sequence labeling via bi-directional lstm-cnns-crf. *CoRR* abs/1603.01354.

[Peng and Dredze 2015] Peng, N., and Dredze, M. 2015. Named entity recognition for chinese social media with jointly trained embeddings. In *EMNLP*.

[Peng and Dredze 2016] Peng, N., and Dredze, M. 2016. Improving named entity recognition for chinese social media with word segmentation representation learning. In *ACL*.

[Peters et al. 2017] Peters, M. E.; Ammar, W.; Bhagavatula, C.; and Power, R. 2017. Semi-supervised sequence tagging with bidirectional language models. In *ACL*.

[Rei 2017] Rei, M. 2017. Semi-supervised multitask learning for sequence labeling. In *ACL*.

[Vaswani et al. 2017] Vaswani, A.; Shazeer, N.; Parmar, N.; Uszkoreit, J.; Jones, L.; Gomez, A. N.; Kaiser, L.; and Polosukhin, I. 2017. Attention is all you need. In *NIPS*.

[Xu et al. 2014] Xu, Y.; Wang, Y. N.; Liu, T.; Liu, J.; Fan, Y.; Qian, Y.; Tsujii, J.; and Chang, E. I.-C. 2014. Joint segmentation and named entity recognition using dual decomposition in chinese discharge summaries. *Journal of the American Medical Informatics Association : JAMIA* 21 e1:e84–92.

[Yang, Zhang, and Dong 2017] Yang, J.; Zhang, Y.; and Dong, F. 2017. Neural word segmentation with rich pretraining. In *ACL*.

[Zhang and Yang 2018] Zhang, Y., and Yang, J. 2018. Chinese NER using lattice LSTM. *CoRR* abs/1805.02023.