2018 IEEE Third International Conference on Data Science in Cyberspace

A Joint Decoding Algorithm for Named Entity Recognition

DongYang Zhao
State Key Laboratory of
High Performance Computing
National University of
Defense Technology
Changsha 410073, China
Email: zhao_yang_1992@126.com

JiuMing Huang
State Key Laboratory of
High Performance Computing
National University of
Defense Technology
Changsha 410073, China
Email: jiuming.huang@qq.com

Yu Luo
and Yan Jia
State Key Laboratory of
High Performance Computing
National University of
Defense Technology
Changsha 410073, China
Email: tel13308423710@126.com
jiayanjy@vip.sina.com

***Abstract*—In the past year, two new models for naming entity recognition have been proposed, namely: Leveraging Linguistic Structures for Named Entity Recognition with Bidirectional Recursive Neural Networks; A FOFE-based Local Detection Approach for Named Entity Recognition and Mention Detection. In this article, we design a co-decoding scheme that can combine the outputs (views) of the two systems to produce an output that is more accurate than the outputs of individual systems.The proposed method has been evaluated in ontonotes 5.0. Experiments show this method surpasses current state-of-the-art on OntoNotes 5.0.**

***Keywords*— FOFE; BRNN-CNN; Joint decoding**

I. Introduction

Named entity recognition is one of the fundamental problems of natural language processing. Its goal is to identify the entity block of the sentence and mark the entity blocks as one of the predefined categories (person, organization, location). Traditionally, the named entity recognition task is considered to be a sequence of labeling tasks, so the general use of recurrent neural network modeling. Chiu et al. [1] used BiLSTM-CNN to achieve the best results two years ago. However, considering named entity recognition as a sequence marking task lacks the structural characteristics of the language itself, but there is a close relationship between the structure of the language and the named entity. Through the analysis found that most named entity chunks are linguistic constituents, such as noun phrases. Therefore, Li et al. [2] Proposed a constituent-based approach for NER, which considers the NER problem as a named entity classification task on every node of a constituency structure. To classify constituents and take into account their structures, they propose BRNN-CNN, a special bidirectional recursive neural network attached with a convolutional network.

In order to identify entities across constituent boundaries(Fig. 1), we use another model for joint decoding. This model is a new local detection method proposed by Xu et al. [3], which rely on the recent fixed-size ordinarily forgetting encoding (FOFE) method to fully encode each sentence fragment and its left/right contexts into a fixed-size representation.

TABLE I
Dataset statistics for OntoNotes 5.0

Split	Token	NE	Constituent
Train	1088503	81828	93.3/97.3
Validate	147724	11066	92.8/97.0
Test	152728	11257	92.9/97.2

This method will locally judge and verify every possible fragment of a sentence for the possible label based on the underlying fragment itself as well as its left and right contexts of the sentence. This method will examine all word segments (up to a certain length) in a sentence one by one. At each time, a word segment will be examined individually based on the underlying segment itself and its left and right contexts of the sentence to determine whether this word segment is a valid named entity.

By jointly decoding the two prediction results, we can On the premise of keeping the result of method one, the entity of cross constituent boundaries can be effectively identified, and the experimental results can be improved. Experiments on ontonotes 5.0(TABLE. I) shows that our method achieves the most state-of-the-art results.

In section II we introduce the related work,In section III we introduce the two methods mentioned above. In section IV we clarify our joint decoding scheme. The section V shows the experimental results, the sixth section to summarize.

II. Related Work

For the sequence labeling task, a recurrent neural network is generally used for modeling. RNNs have shown great success in diverse NLP tasks such as speech recognition (Graves et al. [4]), machine translation (Cho et al. [5]). But RNN has a problem of gradient disappearance [6]. In order to solve this problem, various variants of the recurrent neural network, such as LSTM and GRU [7], have been proposed and have achieved very good results. The long-short term memory (LSTM) unit with the forget gate allows highly non-trivial long-distance dependencies to be easily learned (Gers et al. [8]). Chiu et

978-1-5386-4210-8/18/$31.00 ©2018 IEEE
DOI 10.1109/DSC.2018.00112

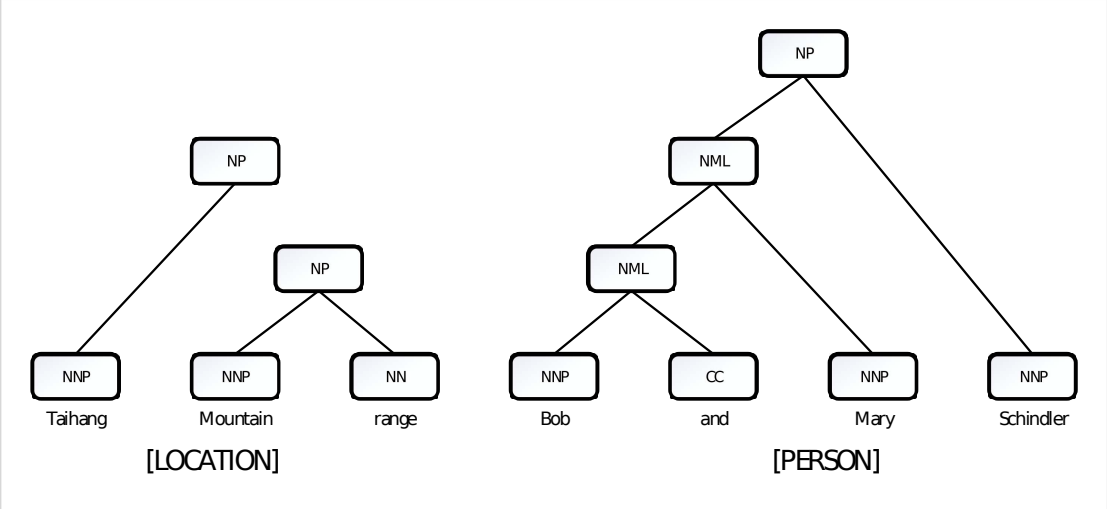

Fig. 1. Two sample named entities that cross constituent boundaries

al. [1] used LSTM-CNN to achieve a good effect, through the use of CNN, character-level features were extracted. At the same time, the use of external dictionary knowledge greatly improved the recognition accuracy. Xu et al. [3] creatively proposes using fixed coding to represent sequence information, which can further to improve the recognition effect while simplifying the model. Li et al. [2] found that most of the named entities are linguistic constituents through the observation of the language structure, and then proposed a named entity method based on the language structure, and achieved the best results currently.

III. Two models

A. the local detection approach

We first introduce the local detection approach, which consists of two components: (1) deep feedforward neural network(Hornik et al. [9]). For classification tasks, the output of the network is normalized to a probability distribution by the softmax function. During training, we can use the backpropagation algorithm to parameterize NN learns to fit the input and the output. The learned NN may be used to generalize and extrapolate to new inputs that have not been seen during training. (2) Fixed-size Ordinally Forgetting Encoding (Zhang et al. [10]). Deep feedforward neural networks require the input size to be fixed and it can not capture the dependencies on the sequence. Therefore, there is a need for a coding scheme that has a fixed length and can capture sequence information. Fixed-size Ordinally Forgetting Encoding exactly meets this requirement. Its specific encoding scheme is as follows:

Give a vocabulary V consisting of $|V|$ distinct words, each word can be represented by an one-hot vector. Let S = w_1, w_2, w_3,....., w_T denote a sequence of T words from V , and e_t be the one-hot vector of the t-th word in S, where $1 \leq t \leq T$.The FOFE of each partial sequence z_t from the first word to the t-th word is recursively defined as:

$$f(x) = \begin{cases} 0, t = 0 \\ \alpha \cdot z_{t-1} + e_t, otherwise \end{cases} \quad (1)$$

where the constant α is called forgetting factor,It is a number between 0 and 1. As you can see, the encoded representation size of this sequence is $|V|$, regardless of the length of the sequence,then we use a projection matrix to compress it. We can theoretically prove the uniqueness of this representation [11] [10] and we can recover the original word sequence from this representation. Using the above method, we can also model on the character level to capture the morphology information. Any word or phrase can be regarded as a sequence of characters. Using the above method, we can derive a fixed-size vector representation of the sequence of characters. Through the above two steps, we can encode the left and right contexts of the word segment and the word segment, and then input them into the depth feedforward neural network to predict the entity type. We use FOFE to capture the following features:

1.Word-level features: Bag-of-word vector of the fragment; FOFE code for left context including the fragment; FOFE code for left context excluding the fragment; FOFE code for right context including the fragment; FOFE code for right context excluding the fragment. all of the above word features are computed for both case-sensitive as well as case-insensitive.These FOFE codes are projected to lower-dimension dense vectors based on two projection matrices, W_s and W_i,which are initialized by GloVe and fine-tuning during model training.

2.Character-level Features: Left-to-right FOFE code of the character sequence of the underlying fragment. Right-to-left FOFE code of the character sequence of the underlying fragment. These case-sensitive character FOFE codes are also

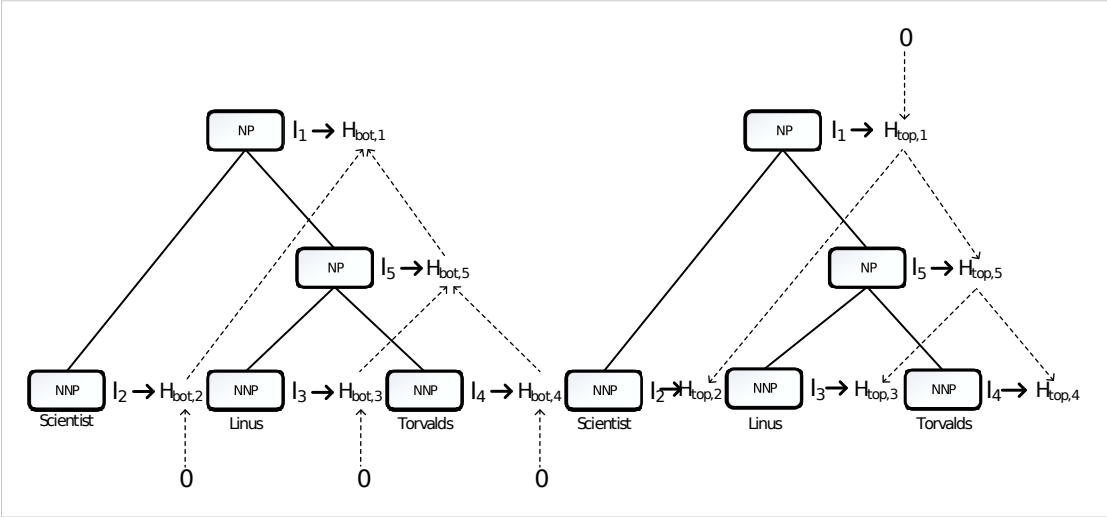

Fig. 2. The bottom-up and top-down hidden layers applied to the binarized tree

projected by another character embedding matrix, which is randomly initialized and fine-tuning during model training.

B. the method of utilizing the linguistic structures of texts

Next, we introduce the method of utilizing BRNN-CNN and the linguistic structures of texts. This method is divided into the following four steps:
Preparing Constituency Structures
Computing Word Embeddings
Computing Hidden Features
Forming Consistent Prediction
Preparing Constituency Structures: For a sentence and its associated constituency parse, we first set four features for each node: a POS, a word, a head and 3-bit vector. Among them semantic head words are determined by a rule-based head finder [12]. The 3-bit vector records if the constituent of a node matches some phrases in each of the three SENNAs([13]) lexicons of persons, organizations, and locations. The system then tries to generate more plausible constituents while preserving linguistic structures by applying a binarization process which groups excessive child nodes around the head children.

Computing Word Embeddings: It is concatenated by two vectors. One is by retrieving a trainable look-up table initialized by GloVe [14]; the other is by using the CNN network for words at the character level [15].

Computing Hidden Features: For each node, calculate its two hidden feature vectors. The hidden feature vector is calculated from the original feature vector. For any node, its original feature vectors are calculated as follows: concatenating the one-hot POS vectors of i, l, r, the head embeddings of i, l, r, the word embedding of i, the lexicon vector of i, and the mean of word embeddings in the sentence. Where l and r are the left sibling and the right sibling of i respectively. The two hidden feature vectors are calculated as follows:

$$H_{bot,i} = ReLU\left(\left(I_i \,||\, \sum_{c \in C} H_{bot,c}\right) W_{bot} + b_{bot}\right) \quad (2)$$

$$H_{top,i} = ReLU\left(\left(I_i \,||\, H_{top,p}\right) W_{top} + b_{top}\right) \quad (3)$$

where W are weight matrices, b are bias vectors,and ReLU(x) = max(0, x).In the above feature vector calculation, zero vectors are used for non-existent variables. The calculation of hidden features can be further extended to multiple layers, with the following formula:

$$H_{t\alpha,i} = ReLU\left(\left(I_i \,||\, H_{t\alpha,p}\right) W_{t\alpha} + b_{t\alpha}\right) \quad (4)$$

$$H_{t\beta,i} = ReLU\left(\left(H_{t\alpha,i} \,||\, H_{t\beta,p}\right) W_{t\beta} + b_{t\beta}\right) \quad (5)$$

where tα represents the first top-down hidden layer and tβ represents the second. Experiments show that the best results can be achieved with three layers. The meanings of hidden features are as follows: The bottom-up direction computes the semantic composition of the subtree of each node, and the topdown counterpart propagates to that node the linguistic structures which contain the subtree. Together, hidden features of a constituent capture its structural linguistic information.As shown in Figure. 2

Forming Consistent Predictions: After calculating the hidden features of a node, calculate the probability distribution for each entity type (including NON-entity) through the output layer. The formula is as follows:

$$O_i = \sigma\left(\left(H_i \,||\, H_l \,||\, H_r\right) W_{out} + b_{out}\right) \quad (6)$$

If you have calculated multiple layers of hidden features, select the last layer as input. In an input sentence, the category corresponding to the highest score of the probability distribution of each constituent is considered as the category of this constituent. If a parent node has been predicted, then the child nodes do not predict, in case of overlapping prediction.

IV. JOINT DECODING SCHEME

The constituent-based approach filters out the other 3% named entities that cross constituent boundaries (Figure. 1), i.e. 3% loss of recall.To identify such named entities, we use the FOFE-based Local Detection method to identify the same sentence, and then we use an entity similarity metric to analyze whether the entities identified by the two methods are consistent. If exactly the same, then the recognition of this sentence is accepted by us. If there are different parts, then verify that each entity identified by the FOFE method is a constituent of a sentence or not. There are three situations,As shown in Figure. 3:

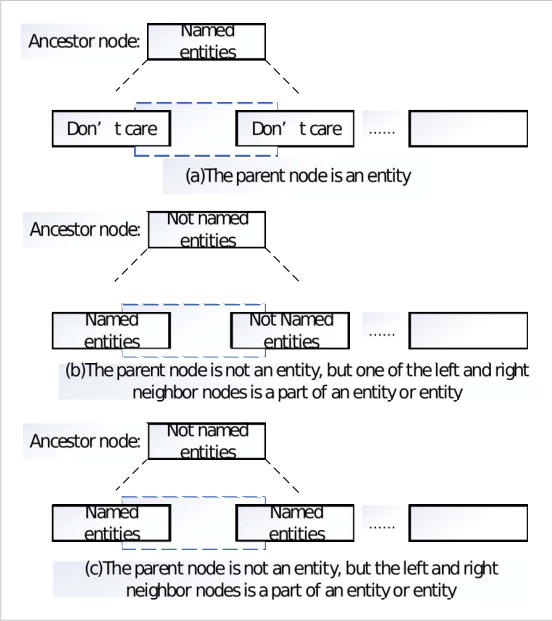

Fig. 3. Three situations arise from the two methods

In the picture above, the blue dotted box is entity identified by the FOFE method. Specifically, for the first case, this fragment is included in an ancestor node identified as an entity in the component analysis tree. As we said before, If a parent node has been predicted, then the child nodes do not predict, in case of overlapping prediction. So we do nothing about this situation. For the second and third cases, if the currently detected entity identified by the FOFE method does not cross the entity identified by the component-based method and exceeds the set threshold, then add this entity to our final recognition result. If the currently detected entity identified by the FOFE method intersects with an entity of the sentence identified by the component-based method and the two types of methods recognize the same entity type, the one with the higher score is retained. If the current detected entity identified by the FOFE method intersects multiple entities of this sentence identified by the component-based method, we do nothing about this situation. For the above second and third situations, we set an acceptance threshold, and when the corresponding score for this entity identified by the FOFE method is greater than this threshold ,then decide what to do based on the above judgment. In this way we identify entities that cannot be identified using linguistic structure methods. And because we set the acceptance threshold, the entity that is added is true with a very high degree of confidence. As a result, our approach increases the recall rate and precision, so the overall F1 value is improved.The entire model used in the experiment is shown in Figure. 4

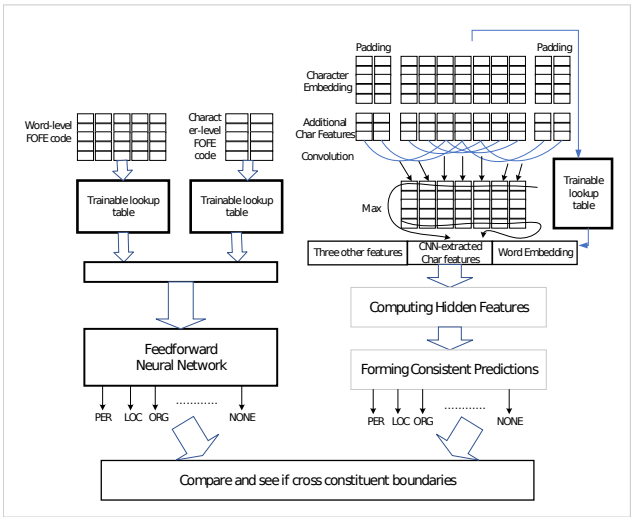

Fig. 4. Joint decoding model for named entity recognition

V. EXPERIMENTAL RESULTS EVALUATION

We use OntoNotes5.0 to evaluate our method. for these two methods, we minimize the cross entropy loss of the softmax class probabilities in Equation 6 by the Adam optimizer ([17]).Some of the hyperparameter settings in the experiment are as follows:Convolution width is 3; Learning rate is 0.01 and dynamically adjusted; Dropout value is 0.5; Mini-batch size is 10; Acceptance threshold is 0.9; Epochs is 20; training lookup table is initialized with GloVe 6B 300d.OntoNotes 5.0 annotates 18 types of named entities for diverse sources of texts,we use the format and the train-validate-test split provided by CoNLL-2012 and auto parses. The corpus-related information is shown in Table. I. The last column shows the percentages of named entities that correspond to constituents of auto parses before and after binarization. From which we can see that the constituent-based approach filters out the other 3% named entities that cross constituent boundaries (Figure. 1), ie 3% loss of recall.Table. II compares the results of our method with those of other methods. As can be seen from the table, our method is significantly superior to Chiu's method. In addition, Peng-Li's method is slightly lower than the experimental results of our method, because our joint decoding algorithm has played a role. Table. III shows the F1 scores on different data sources. From the table, we can

TABLE II
EXPERIMENT RESULTS ON WHOLE DATASET

		Validation			Test		
Model	Parser	Precision	Recall	F1	Precision	Recall	F1
Durrett and Klein(2014) [16]	-	-	-	-	85.22	82.89	84.04
chiu and nichols(2016) [1]	-	-	-	-	-	-	86.41
BRNN-CNN [2]	auto	85.5	84.7	85.08	88.0	86.5	87.21
BRNN-CNN [2]	gold	86.6	87.0	86.77	88.9	88.9	88.92
our method	auto	85.48	85.0	85.24	87.91	86.63	87.32

TABLE III
F1 SCORES ON DIFFERENT DATA SOURCES

Model	BC	BN	MZ	NW	TC	WB
Test set size(tokens)	32576	23557	18260	51667	11015	19348
Test set size(entities)	1697	2184	1163	4696	380	1137
Finkel and Manning (2009)	78.66	87.29	82.45	85.50	67.27	72.56
Durrett and Klein (2014) [16]	78.88	87.39	82.46	87.60	72.68	76.17
BLSTM-CNN + emb + lex [1]	85.23	89.93	84.45	88.39	72.39	78.38
BRNN-CNN-auto [2]	85.98	90.96	84.93	89.18	73.18	80.39
our method	86.05	91.09	85.12	89.23	73.21	80.51

see that for different data sources, the accuracy of our method has been improved to varying degrees.

VI. CONCLUSION

In this paper, we propose a joint decoding algorithm that complements a constituent-based approach with a local detection method that allows our named entity recognition results to be higher on the basis of a constituent-based approach. The experimental results of our method on the ontonotes 5.0 dataset show that this joint decoding algorithm can be more efficient than either method alone.And the recognition result of this method is partly better than the previous method using external dictionaries and external knowledge. However, this method takes more time and one of the future directions of work is to further improve the efficiency of the algorithm and reduce the consumption time.

ACKNOWLEDGMENT

This work is supported by the National Key Research and Development Program No.2016QY03D0603.

REFERENCES

[1] J. P. C. Chiu and E. Nichols, "Named entity recognition with bidirectional lstm-cnns," *Computer Science*, 2015.

[2] P.-H. Li, R.-P. Dong, Y.-S. Wang, J.-C. Chou, and W.-Y. Ma, "Leveraging linguistic structures for named entity recognition with bidirectional recursive neural networks," in *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*, 2017, pp. 2664–2669.

[3] M. Xu, H. Jiang, and S. Watcharawittayakul, "A local detection approach for named entity recognition and mention detection," in *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, vol. 1, 2017, pp. 1237–1247.

[4] A. Graves, A. R. Mohamed, and G. Hinton, "Speech recognition with deep recurrent neural networks," vol. 38, no. 2003, pp. 6645–6649, 2013.

[5] K. Cho, B. Van Merrienboer, D. Bahdanau, and Y. Bengio, "On the properties of neural machine translation: Encoder-decoder approaches," *Computer Science*, 2014.

[6] J. F. Kolen and S. C. Kremer, "Gradient flow in recurrent nets: The difficulty of learning longterm dependencies," vol. 28, no. 2, pp. 237–243, 2001.

[7] K. Cho, B. V. Merrienboer, C. Gulcehre, D. Bahdanau, F. Bougares, H. Schwenk, and Y. Bengio, "Learning phrase representations using rnn encoder-decoder for statistical machine translation," *Computer Science*, 2014.

[8] F. A. Gers, J. A. Schmidhuber, and F. A. Cummins, "Learning to forget: Continual prediction with lstm," in *Artificial Neural Networks, 1999. ICANN 99. Ninth International Conference on*, 2002, p. 2451.

[9] K. Hornik, "Approximation capabilities of multilayer feedforward networks," *Neural networks*, vol. 4, no. 2, pp. 251–257, 1991.

[10] Zhang, "The fixed-size ordinally-forgetting encoding method for neural network language models," in *Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, vol. 2, 2015, pp. 495–500.

[11] S. Zhang, H. Jiang, M. Xu, J. Hou, and L. Dai, "A fixed-size encoding method for variable-length sequences with its application to neural network language models," *Computer Science*, 2015.

[12] M. Collins, "Head-driven statistical models for natural language parsing," *Computational linguistics*, vol. 29, no. 4, pp. 589–637, 2003.

[13] R. Collobert, J. Weston, L. Bottou, M. Karlen, K. Kavukcuoglu, and P. Kuksa, "Natural language processing (almost) from scratch," *Journal of Machine Learning Research*, vol. 12, no. Aug, pp. 2493–2537, 2011.

[14] J. Pennington, R. Socher, and C. Manning, "Glove: Global vectors for word representation," in *Proceedings of the 2014 conference on empirical methods in natural language processing (EMNLP)*, 2014, pp. 1532–1543.

[15] Y. Kim, Y. Jernite, D. Sontag, and A. M. Rush, "Character-aware neural language models." in *AAAI*, 2016, pp. 2741–2749.

[16] G. Durrett and D. Klein, "A joint model for entity analysis : Coreference , typing , and linking," 2014.

[17] D. Kingma and J. Ba, "Adam: A method for stochastic optimization," *Computer Science*, 2014.