DUTIR at the BioCreative V.5.BeCalm Tasks: A BLSTM-CRF Approach for Biomedical Entity Recognition in Patents

Ling Luo, Pei Yang, Zhihao Yang*, Hongfei Lin, Jian Wang

College of Computer Science and Technology,
Dalian University of Technology, Dalian, China 116024

{lingluo, yangperasd}@mail.dlut.edu.cn;
{yangzh, hflin, wangjian}@dlut.edu.cn

Abstract. Patents contain the significant amount of information. Biomedical text mining has received much attention in patents recently, especially in the medicinal chemistry domain. The BioCreative V.5.BeCalm tasks focus on biomedical entities recognition in patents. This paper describes our method used to create our submissions to the Chemical Entity Mention recognition (CEMP) and Gene and Protein Related Object recognition (GPRO) subtasks. In our method, a bidirectional Long Short-Term Memory with a conditional random field layer (BLSTM-CRF) is employed to recognize biomedical entities from patents. Our best CEMP submission achieves an F-score of 90.42% and our best GPRO submission with type 1 achieves an F-score of 79.19%.

Keywords. Patents; Biomedical Entity Recognition; Deep Learning; Long Short-Term Memory; Conditional Random Field

1 Introduction

Biomedical named entity recognition (NER) is a fundamental step for the complex biomedical natural language processing (NLP) tasks (such as entity relation extraction). In the previous BioCreative tasks [1-3], various approaches have been proposed to recognize biomedical entities from the scientific literature. In addition to the scientific literature, patents are another important source, and they contain a wealth of useful biomedical information. Therefore, automatic extraction the information contained in patents has received much attention, especially in the medicinal chemistry domain. Among others, automatic biomedi-

cal entity recognition from medicinal chemistry patents has become an important research task [4].

To promote the development of NER systems, the BioCreative V.5, a major challenge event in biomedical natural language processing, proposes the BeCalm tasks. This challenge included three individual subtasks. The first subtask is technical interoperability and performance of annotation servers (TIPS). This task focuses on the technical aspects of the evaluation of continuous text Annotation Servers for NER. The second subtask is chemical entity mention recognition (CEMP). This task requires the detection of chemical named entity mentions from patent titles and abstracts. The third subtask is gene and protein related object recognition (GPRO). The task requires the detection of gene and protein related objects mentions from patent titles and abstracts. We participated in the CEMP and GPRO subtasks and our submissions to the two subtasks are created by the deep learning model, which is a bidirectional Long Short-Term Memory with a conditional random field layer (BLSTM-CRF). Our methods and results are presented in the following sections.

2 Discussion

Similar to many NER tasks, we modeled the CEMP and GPRO tasks as a sequence labeling problem. We used the BIO (Begin, Inside, Outside) tagging scheme since it achieves better performance than BIOES tagging scheme in our experiments. For the challenge, we presented the neural network architecture, a bidirectional Long Short-Term Memory with a conditional random field layer (BLSTM-CRF), to recognize biomedical entities from patents. The processing flow of our method is shown in Figure 1. Firstly, some preprocessing steps including text cleaning, sentence splitting and tokenization are performed. Secondly, a word embedding is learned with large amounts of unlabeled data by the word2vec tool. Moreover, we induce the character embedding and linguistic feature embeddings. Then with the embeddings as input, a BLSTM-CRF model is trained by the annotated training set. Finally, some post-processing steps including tagging consistency, abbreviation resolution and bracket balance are employed. The process is described in details in the following sections.

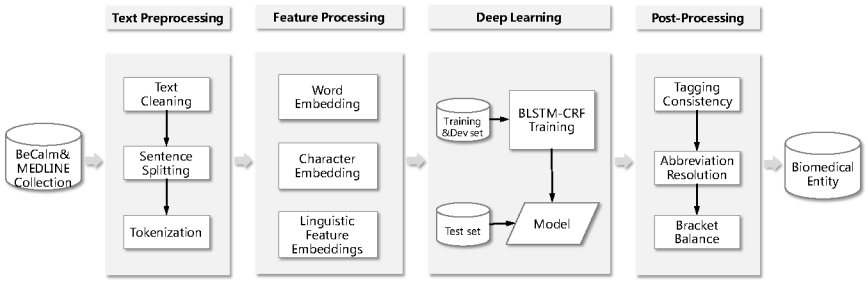

Figure 1. The processing flowchart of our method

2.1 Features

Word embedding and character embedding are widely used in the field of NLP, especially based on the deep learning methods. We used them as the features of our baseline. Moreover, to investigate the effects of traditional linguistic features (such as part of speech (POS) and chunking), these linguistic features are added into the baseline as different runs. All feature embeddings are parameters of the model, and they can be optimized when the model is trained. Details of each of features are presented as follows.

2.1.1 Word Embedding

Word embedding, also known as distributed word representation, can capture both the semantic and syntactic information of words from a large unlabeled corpus and has attracted considerable attention from many researchers [5]. Compared with the bag-of-words (BOW) representation, word embedding is low-dimensional and dense. In recent years, several models, such as word2vec [6] and GloVe [7], have been proposed and widely used in the field of NLP. To achieve a high-quality word embedding, we downloaded a total of 1,918,662 MEDLINE abstracts from the PubMed website with the query string "drug" as the unlabeled data. Then the data and all datasets (The training set comprises a total of 21,000 abstracts, and the test set comprises a total of 9,000 abstracts.) provided in the BeCalm tasks were used to train 50-dimensional word embedding by the word2vec tool as pre-trained word embedding.

2.1.2 Character Embedding

In addition to the word embedding, character-level features in a name contain rich structure information of the entity. These features (such as character n-grams, prefixed and suffixes) are commonly employed in the current NER methods [8]. Unlike the previous traditional methods in which character features are based on hand-engineering, character embedding can be learned while training. Character embedding has been found useful for many NLP tasks. They can not only learn interior representations of the names, but also alleviate the out-of-vocabulary problem. In our method, a character lookup table which contains a 25-dimensional embedding for every character is initialized randomly. Then the character embedding corresponding to every character in a word is given in both direct and reverse orders to a bidirectional LSTM. At last, the concatenation of the forward and backward representations from the bidirectional LSTM is used as the character-level feature of the word.

2.1.3 Linguistic Feature Embedding

Due to the complexity of the natural language, some linguistic features are often employed in traditional machine learning methods [9]. We also explored the effect of linguistic features (such as POS and chunking). The POS information and chunking information of each word were generated by the GENIA tagger [10]. In addition, named entity tags information generated by the GENIA tagger was also used as a feature. The dimensions of the POS, chunking and named entity tags embedding are 25, 10 and 5, respectively. They were initialized randomly.

2.2 BLSTM-CRF Model

The architecture of our model, a bidirectional Long Short-Term Memory with a conditional random field layer (BLSTM-CRF), is illustrated in Figure 2.

Recurrent neural networks (RNNs) are a family of neural networks for processing sequential data. Giving a sequence of vectors $(\mathbf{x}_1, \mathbf{x}_2, ..., \mathbf{x}_t, ..., \mathbf{x}_n)$ as input, they return another corresponding sequence $(\mathbf{h}_1, \mathbf{h}_2, ..., \mathbf{h}_t, ..., \mathbf{h}_n)$. The current state $\mathbf{h}_t$ is generated from the input $\mathbf{x}_t$ and the state $\mathbf{h}_{t-1}$ that is passed forward though time. However, traditional RNNs have the mathematical challenge of learning long-term dependencies. The main problem is that gradients propagated over

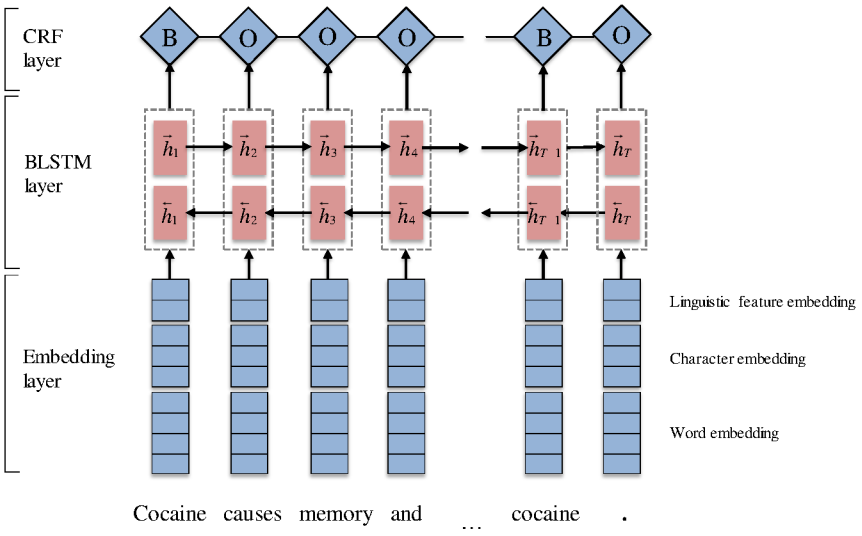

Figure 2. The architecture of BLSMT-CRF model

many stages tend to vanish. When the sequence is long, traditional RNNs are difficult to work well. To alleviate this problem, Long Short-Term Memory (LSTM) [11] is designed by incorporating a memory cell with the gating mechanism and has been shown to capture long-range dependencies. Therefore, LSTM is applied in our method. LSTM memory cell is implemented as the following:

$$\mathbf{i}_t = \sigma(\mathbf{W}_{xi}\mathbf{x}_t + \mathbf{W}_{hi}\mathbf{h}_{t-1} + \mathbf{W}_{ci}\mathbf{c}_{t-1} + \mathbf{b}_i) \quad (1)$$

$$\mathbf{c}_t = (1-\mathbf{i}_t)\odot\mathbf{c}_{t-1} + \mathbf{i}_t \odot \tanh(\mathbf{W}_{xc}\mathbf{x}_t + \mathbf{W}_{hc}\mathbf{h}_{t-1} + \mathbf{b}_c) \quad (2)$$

$$\mathbf{o}_t = \sigma(\mathbf{W}_{xo}\mathbf{x}_t + \mathbf{W}_{ho}\mathbf{h}_{t-1} + \mathbf{W}_{co}\mathbf{c}_t + \mathbf{b}_o) \quad (3)$$

$$\mathbf{h}_t = \mathbf{o}_t \odot \tanh(\mathbf{c}_t) \quad (4)$$

where σ is the element-wise sigmoid function, and $\odot$ is the element-wise product. $\{\mathbf{W}_{xi}, \mathbf{W}_{hi}, \mathbf{W}_{ci}, \mathbf{W}_{xc}, \mathbf{W}_{hc}, \mathbf{W}_{xo}, \mathbf{W}_{ho}, \mathbf{W}_{co}\}$ is the weight matrix set. $\{\mathbf{b}_i, \mathbf{b}_c, \mathbf{b}_o\}$ is the bias vector set.

However, the LSTM's hidden state $\mathbf{h}_t$ only takes the information from the left context of the sequence at every time t. An elegant solution is a bidirectional LSTM (BLSTM) [12]. In the BLSTM architecture, a forward LSTM computes a representation $\overrightarrow{\mathbf{h}_t}$ of the sequence

from left to right, and another backward LSTM computes a representation $\overleftarrow{\mathbf{h}_t}$ of the same sequence in reverse. These two distinct networks use different parameters, and then the representation of a word is obtained by concatenating its left and right context representations, i.e. $\mathbf{h}_t = [\overrightarrow{\mathbf{h}_t}; \overleftarrow{\mathbf{h}_t}]$. The representation can make use of rich context information in predicting the current tag.

In the NER tasks, the output labels have strong dependencies. In addition to information of the word itself and the context, the entity tag of the word is also decided by the context tags information of the word. For example, in a reasonable entity tag sequence, the tag "I" generally appears after the tag "B", but it does not appear after the tag "O". However, the above-mentioned BLSTM model only uses the $\mathbf{h}_t$ to make independent tagging decisions for each output. Therefore, instead of modeling tagging decisions independently, the CRF layer are added after the BLSTM layer to decode the best tag path in all possible tag paths.

In our method, the NER task is to assign an entity tag to every word in a sentence. Firstly, the word embedding, character embedding and linguistic feature embedding are concatenated as input to feed the BLSTM layer. Then the output of BLSTM layer is fed into the CRF layer. At last, the CRF layer decodes the best tag path.

To be more specific, we introduce a tagging transition matrix $\mathbf{A}$, where $A_{i,j}$ represents the score of transition from tag i to tag j in successive words. This transition matrix will be trained as the parameter of model. We define θ as the set of parameters for the original BLSTM, and $\theta' = \theta \cup \{A_{i,j} \forall i, j\}$ as the set of all parameters for the BLSTM-CRF model. For an input sentence $[x]_1^T$ where T is the length of the sentence, we consider $f_\theta([x]_1^T)$ to be the matrix of scores output by the BLSTM network. The element $[f_\theta]_{i,t}$ of the matrix is the score of the i^{th} tag of the t^{th} word in the sentence. The score of the sentence along with a tag path $[i]_i^T$ is then given by the sum of transition scores and network scores:

$$S([x]_1^T, [i]_1^T, \theta') = \sum_{t=1}^{T} (A_{[i]_{t-1},[i]_t} + [f_\theta]_{[i]_t,t}) \tag{5}$$

Then we use a softmax function to yield the conditional probability of the path $[y]_1^T$ by normalizing the above score over all possible tag paths $[j]_1^T$:

$$p([y]_1^T \mid [x]_1^T, \theta') = \frac{e^{S([x]_1^T, [y]_1^T, \theta')}}{\sum_j e^{S([x]_1^T, [j]_1^T, \theta')}} \quad (6)$$

During the training phase, we maximize the log-probability of the correct tag sequence:

$$\log P([y]_1^T \mid [x]_1^T, \theta') = S([x]_1^T, [y]_1^T, \theta') - \log \sum_{\forall [j]_1^T} e^{S([x]_1^T, [j]_1^T, \theta')} \quad (7)$$

Stochastic gradient descent (SGD) is used to optimize the model parameters. At inference time, we predict the best tag path that obtains the maximum score given by:

$$\arg\max_{[j]_1^T} S([x]_1^T, [j]_1^T, \theta') \quad (8)$$

This can be computed using dynamic programming, and the Viterbi algorithm [13] is chose for this inference.

2.3 Post-Processing

In our method, we employed several common post-processing steps including tagging consistency, abbreviation resolution and bracket balance.

- If the number of a word sequence tagged by our model as a biomedical entity exceeds 50% of the total number of the sequence in a document (title and abstract), all instances of the word sequence will be tagged as an entity. For example, if our model found three chemical entities of "flunarizine hydrochloride" and missed out two other entities of "flunarizine hydrochloride" in a document, the missed entities would be retrieved.
- For abbreviation resolution, all local abbreviation definitions, such as "gamma-aminobutyric acid (GABA)", will be found. If the abbreviation (i.e. GABA) in the long form was tagged by our model, then all instances of the abbreviation would be tagged in a document.
- While there are some entities with unbalanced brackets (such as parenthesis, square brackets and curly brackets), we attempted to balance the brackets by adding or removing characters to the right or left of the entity. For example, if "A-(X1-N" (the next characters in the text

are “O2)”) was tagged as an entity by our model, then the entity would be extended to include the right parenthesis (i.e. “A-(X1-NO2)”).

2.4 Training Procedure

In our method, the parameters of the model in the word embedding are initialized with pre-trained word embeddings and other parameters are initialized at random from a uniform distribution. Then all parameters are optimized using SGD to maximize the log-probability of the correct tag sequence. In addition, several hyper-parameters need to be determined in our model. We tuned the hyper-parameters on the validation set by random search. Our models are implemented using open-source deep learning library Theano and trained on a NVIDIA Tesla K40 GPU.

3 Results

The organizers of the BioCreative V.5.BeCalm tasks provided a corpus including the training and test sets. The training set comprises a total of 21,000 manually annotated documents (title and abstract), and test set comprises a total of 9,000 unannotated documents. Annotations for the CEMP task are provided in seven classes: systematic, identifiers, formula, trivial, abbreviation, family and multiple. In the case of the GPRO task, the annotations are divided in two groups: type 1, covering GPRO mentions that can be normalized to a database record (including the following classes: nested mentions, identifier, full name and abbreviation); and type 2, covering those GPRO mentions that in principle cannot be normalized to a unique bio-entity database record (including the following classes: no class, sequence, family and multiple).

3.1 CEMP Task Results

In the CEMP task, we randomly selected the 10% of the training set as the development set to tune the hyper-parameters. Although seven annotation classes for chemical entities are provided, discrimination between the classes is not an objective of the task. Therefore, in our method, all classes are grouped into a single one. The results of our submitted runs on our development set and official results of the runs on the test set are shown in Table 1.

Our submitted five runs are based on the following configurations.

Table 1. CEMP task results

Runs	Development set			Test set		
	Precision	Recall	F-score	Precision	Recall	F-score
1	87.30	92.28	89.72	88.32	**92.62**	**90.42**
2	-	-	-	84.21	89.42	86.74
3	87.21	**92.58**	89.81	88.42	92.13	90.24
4	**88.12**	91.7	**89.87**	**89.38**	91.03	90.20
5	87.54	91.23	89.35	89.18	91.35	90.25

- Run 1: our BLSM-CRF model with word embedding and char embedding as inputs.
- Run 2: using the same features and model as Run 1, but the development set was added into the training set (for the other runs, the development set is not used for training). And according to our experience, we stopped the model training at 20 iterations.
- Run 3: like Run 1, but uses the additional chunking feature.
- Run 4: like Run 1, but uses the additional POS feature.
- Run 5: like Run 1, but all features were used.

The results show that there is no significant difference among F-scores of these runs except Run 2. The reason is that, since the development set was added into the training set, Run 2 was stopped training at 20 iterations according to our experience. Probably we stopped the model training prematurely so that the model parameters could not be optimized. In contrast, the other runs were trained by early stopping strategy [14] on the development set. Therefore, the performance of Run 2 is worse than those of the other runs.

In addition, on both the development and test sets, no significant improvement was observed with the additional features. The plausible reason is that the deep neural network itself has learned sufficient higher and abstract features automatically from the word and character embeddings.

3.2 GPRO Task Results

In the GPRO task, only 5,795 documents in the training set contain annotated gene entities and the rest 15,205 documents do not contain

Table 2. GPRO task results

Runs	Results of GPRO type 1 on the development set			Results of GPRO type 1 on the test set			Results of GPRO type 1 and 2 on the test set		
	P	R	F	P	R	F	P	R	F
1	67.97	**86.06**	75.95	74.61	**83.70**	78.89	80.12	61.14	69.35
2	64.26	84.65	73.06	73.71	82.51	77.86	81.56	**64.89**	**72.28**
3	63.16	85.69	72.72	72.42	83.36	77.51	79.21	63.89	70.73
4	-	-	-	**78.20**	76.81	77.50	**83.67**	54.10	65.71
5	**70.84**	83.76	**76.76**	76.65	81.91	**79.19**	81.06	57.89	67.54

P denotes precision, R denotes recall and F denotes F-score.

gene entities. In our experiments, to explore the effectiveness of the documents without annotated gene entities, two corpora were used to train the different runs. One consists of the documents data in the original training set ($Data_{GPRO\_ori}$); the other consists of all documents with annotated gene entities and the documents of the same number (5,795) without gene entities in the original training set ($Data_{GPRO\_balance}$). We randomly selected the 10% of the two corpora as the corresponding development set, respectively and the similar models in the CEMP task were employed for the GPRO task. However, we mistakenly thought only entities that can be mapped to an identifier (type 1) are evaluated like the GPRO subtask in the BioCreative V does [4]. Therefore, our final results only provide the type 1 entities, and the type 2 entities are ignored. Finally, five runs based on the following configurations were chosen to submit.

- Run 1: our BLSM-CRF model with word embedding and char embedding as inputs. Type 1 and 2 entities were treated as two distinct classes, and the model was trained with the $Data_{GPRO\_balance}$.
- Run 2: the result was produced by the ensemble of two models. Firstly, model 1 was trained with the $Data_{GPRO\_ori}$ using the same features and model as Run 1, but type 1 and 2 entities were treated as a single class. Then the type 2 entities recognized by the model 2 (i.e. the model for Run 3) were removed from all entities recognized by the model 1.
- Run 3: like Run 1, but uses the additional chunking feature.
- Run 4: the same features and model as Run 1, but the model was trained with the $Data_{GPRO\_ori}$ and the development set was added. According to our experience, we stopped the model training at 20 iterations.

- Run 5: like Run 1, but all features were used.

The results of different runs on our development set and official results of all runs on the test set are shown in Table 2. Among others, Run 5 with all features achieves the best performance for GPRO type 1 on the test set. The reason is that the additional linguistic features can help model distinguish type 1 genes from type 2 genes. Unfortunately, because our final results only provide the recognitions of the type 1 entities, the recalls significantly decrease on the test set when both type 1 and type 2 entities are evaluated.

4 Acknowledgment

This work is supported by the grants from the Natural Science Foundation of China (No. 61272373, 61572102 and 61572098), Trans-Century Training Program Foundation for the Talents by the Ministry of Education of China (NCET-13-0084), the Fundamental Research Funds for the Central Universities (No. DUT14YQ213) and the Major State Research Development Program of China (No. 2016YFC0901902).

REFERENCES

1. Smith L, Tanabe LK, nee Ando RJ et al. Overview of BioCreative II gene mention recognition, Genome biology 2008;9:S2.
2. Krallinger M, Leitner F, Rabal O et al. CHEMDNER: The drugs and chemical names extraction challenge, Journal of cheminformatics 2015;7:S1.
3. Wei C-H, Peng Y, Leaman R et al. Overview of the BioCreative V chemical disease relation (CDR) task. In: Proceedings of the fifth BioCreative challenge evaluation workshop. 2015, p. 154-166.
4. Krallinger M, Rabal O, Lourenço A et al. Overview of the CHEMDNER patents task. In: Proceedings of the fifth BioCreative challenge evaluation workshop. 2015, p. 63-75.
5. Lai S, Liu K, Xu L et al. How to Generate a Good Word Embedding?, arXiv preprint arXiv:1507.05523 2015.
6. Mikolov T, Sutskever I, Chen K et al. Distributed representations of words and phrases and their compositionality. In: Advances in neural information processing systems. 2013, p. 3111-3119.
7. Pennington J, Socher R, Manning CD. Glove: Global vectors for word representation, Proceedings of the Empiricial Methods in Natural Language Processing (EMNLP 2014) 2014;12:1532-1543.
8. Liu S, Tang B, Chen Q et al. Drug name recognition: Approaches and resources, Information 2015;6:790-810.

9. Eltyeb S, Salim N. Chemical named entities recognition: a review on approaches and applications, Journal of cheminformatics 2014;6:17.
10. Tsuruoka Y. GENIA tagger: Part-of-speech tagging, shallow parsing, and named entity recognition for biomedical text, Available at: www-tsujii. is. su-tokyo. ac. jp/GENIA/tagger 2006.
11. Hochreiter S, Schmidhuber J. Long short-term memory, Neural computation 1997;9:1735-1780.
12. Graves A, Schmidhuber J. Framewise phoneme classification with bidirectional LSTM and other neural network architectures, Neural Networks 2005;18:602-610.
13. Viterbi A. Error bounds for convolutional codes and an asymptotically optimum decoding algorithm, IEEE transactions on Information Theory 1967;13:260-269.
14. Prechelt L. Automatic early stopping using cross validation: quantifying the criteria, Neural Networks 1998;11:761-767.