A Boundary Assembling Method for Nested Biomedical Named Entity Recognition

YANPING CHEN1,2, YING HU1, YIJING LI1, RUIZHAND HUANG1,2, YONGBIN QIN1,2, YUEFEI WU1,3, QINGHUA ZHENG4, (Member, IEEE), AND PING CHEN3, (Member, IEEE)

1Guizhou Provincial Key Laboratory of Public Big Data, College of Computer Science and Technology, Guizhou University, Guiyang 550025, China
2Key Laboratory of Intelligent Medical Image Analysis and Precise Diagnosis of Guizhou Province, Guizhou University, Guiyang 550025, China
3Department of Computer Science and Technology, Xi’an Jiaotong University, Xi’an 710049, China
4Department of Computer Science, University of Massachusetts Boston, Boston, MA 02125, USA

Corresponding author: Yanping Chen (ypench@gmail.com)

This work was supported in part by the National Natural Science Foundation of China through the Joint Funds under Grant U1836205, in part by the National Natural Science Foundation of China through the Major Research Program under Grant 91746116, in part by the National Natural Science Foundation of China under Grant 62066007 and Grant 62066008, in part by the Major Special Science and Technology Projects of Guizhou Province under Grant [2017]3002, and in part by the Key Projects of Science and Technology of Guizhou Province under Grant [2020] 1Z055.

ABSTRACT Biomedical named entity recognition (BNER) is an important task in biomedical natural language processing, in which neologisms (new terms, words) are coined constantly. Most of the existing work can only identify biomedical named entities with flattened structures and ignore nested biomedical named entities and discontinuous biomedical named entities. Because biomedical domains often use nested structures to represent semantic information of named entities, existing methods fail to utilize abundant information when processing biomedical texts. This paper focuses on identifying nested biomedical named entities using a boundary assembly (BA) model, which is a cascading framework consisting of three steps. First, start and end named entity boundaries are identified and then assembled into named entity candidates. Finally, a classifier is implemented for filtering false named entities. Our approach is effective in handling nesting and discontinuous problems in biomedical named entity recognition tasks. It improves the performance considerably, achieving an F1-score of 81.34% on the GENIA dataset.

INDEX TERMS Biomedical nested named entity recognition, deep learning, information extraction.

I. INTRODUCTION

With rapid progress of biomedical research, the biomedical literature is vast. For example, PubMed, a search engine accessing the MEDLINE database, contains more than 29 million publications. Manually keeping track of biomedical literature is almost impossible. Biomedical information extraction (BioIE) directly identifies structured knowledge from plain texts. It provides a content-oriented approach for processing biomedical texts. Because BioIE only focuses on specific information and ignores unrelated content, it can avoid errors caused by a “deep” analysis, e.g., sentence parsing or understanding. Therefore, BioIE has received extensive attention.

BioIE has defined three typical tasks: named entity recognition (e.g., cells, genes and proteins), relation recognition (e.g., implicit renaming, protein encoding and biological proof) and event recognition (e.g., transcription, regulation and binding) [30], [40], [41]. Among them, biomedical NE recognition is the most fundamental task. It is the key to automatically support biomedical literature processing.

The task of biomedical named entity (NE) recognition is usually modeled as a sequence labeling task. It gives each word a tag to indicate its semantic role, e.g., start, inside or outside of an NE (the S-I-O encoding). Given a sentence, sequence models (e.g., the hidden Markov model (HMM) [11], the conditional random field model (CRF) [15] and the long short-term memory (LSTM) [49]) are often adopted to generate a maximized label sequence. For example, the sentence “The effect of Toremifene on the expression of some genes in human mononuclear cells.” can be labeled as “O-O-O-S-O-O-O-O-O-S-I-I”, in which two NEs...
are recognized ("Toremifene" and "human mononuclear cells").

The sequence model assumes that all NEs in a sentence have a flattened structure. This assumption often fails in biomedical information extraction, in which nested and discontinuous structures are widely adopted to name a biomedical entity. An example of nested and discontinuous NEs is given in Figure 1.

The nested structure is effective in expressing semantic meaning (e.g., affiliation, ownership, and hyponymy) of a biomedical NE. It is widely used to name technical terms, such as DNA, protein and cell type. For example, "immunoglobulin enhancer" is a biomedical NE, where "immunoglobulin" is a protein representing the type of the enhancer. Therefore, recognizing nested NEs is helpful for understanding a term from its typeface. Many annotation corpora allow nested NEs, e.g., GENIA [2], BioInfer [3] and ACE [4], GermEval [5], archive text [6] and clinical reports [7]. In the GENIA and ACE corpora, approximately 35.27% and 33.90% NEs are nested (or discontinuous), respectively. Simply ignoring this phenomenon will lose a great deal of information in the biomedical literature.

Recognizing nested NEs is helpful for capturing more fine-grained semantic information from biomedical text. This task has received extensive attention. However, its performance still has room for improvement. One main problem with existing work is that sequence models are not effective in recognizing nested and discontinuous NEs. As a compromise, three strategies can be adopted for a sequence model to recognize nested NEs: layering approach, cascading approach and joint approach [8]. The layering approach recognizes nested entities layer by layer from inside to outside. The cascading approach recognizes every type of NE by an independent classifier. The joint approach uses structured tags to indicate nested entities. For example, "S-DNA + S-Protein" represents that a labeling unit is the start of a DNA name and a protein name simultaneously.

These strategies suffer from several problems. 1) Layering and cascading approaches cannot make full use of annotated data. Furthermore, they are still unable to recognize nested NEs in the same layer or the same type. 2) The joint approach increases the number of NE types, which hurts performance. 3) Because a sequence model requires a flattened input, some labels must be changed to obtain a flattened structure. Incorrect labels will disrupt the performance. 4) The task of NE recognition is implemented at the sentence level. A sentence usually contains a limited number of words, which causes the sparsity feature problem. 5) Sequence models often adopt the first-order Markov dependency, which cannot make use of global features [10].

In our previous work [1], [9], a boundary assembling (BA) method was proposed to support NE recognition. Instead of modeling the task as a sequence labeling process, it uses a cascading framework to recognize nested NEs. In this framework, NEs are not seen as unitary units. They are processed as the combination of entity boundaries (start and end). This process of NE recognition is divided into three steps: boundary detecting, boundary assembling, and entity discriminating. In the first step, boundaries are detected. Then, they are assembled into entity candidates, where nestification is allowed. Finally, in the discrimination step, another classifier is used to select true cases. This framework has five advantages. 1) Entity boundaries have a smaller granularity, which is not dependent on any NLP tasks. 2) Compared to NEs, boundaries are more unambiguous and clear. They are easier to recognize. 3) All entity boundaries share similar contexts. They can make full utilization of annotated data. 4) Because boundaries are no longer nested, there is no need to change conflicted labels for a flattened structure. 5) The BA method can make better use of global features. After NE candidates are generated, global features can be extracted to support the NE recognition task.

The original BA method was designed to recognize Chinese NEs, where a Chinese sentence consists of Chinese characters (or Hanzi) with the same unit length. It is convenient to obtain the semantic and position information of an NE boundary. English is an alphabetic language that differs from the Chinese hieroglyphic writing system. In this paper, transforming the BA method for English biomedical NE recognition faces several challenges. First, words are the basic units to compose English sentences. English words are various in length, and hyphens are widely used in biomedical literature. Therefore, entity boundaries are vaguer than Chinese characters. Second, English words are composed of morphemes that are informative. Embedding each word into a vector for predicting entity boundaries cannot make full use of semantic information in a word. Third, English biomedical literature contains a large number of abbreviations. Rare semantic features can be extracted from the abbreviation for boundary detection. Fourth, biomedical NEs have been annotated with discontinuous structures. It is difficult to assemble them as NE candidates.

In this paper, the BA method is redesigned for English biomedical NE recognition. In addition to a word-based sequence model, an RNN layer is adopted to capture character-level features in every English word. It is effective for capturing semantic information of morphemes for NE recognition. In the entity discriminating step, instead of the multILSTM model, a convolutional neural network is designed to make a final decision. The main contributions of this paper include the following:

1) In this paper, two neural network structures are designed for detecting biomedical NE boundaries and predicting NE candidates respectively.
2) In addition to the 5 NE-type settings used in the GENIA corpus, all NEs are considered, including discontinuous biomedical NEs that are not considered in other existing methods.

3) Compared with the state-of-the-art nested NE recognition methods, our approach shows impressive improvements. We achieved an 80.57% F1-score on the GENIA dataset and an 83.36% F1-score on the ACE2005 dataset.

The rest of this paper is organized as follows. Related work about nested named entity recognition is given in Section II. Information about the GENIA dataset is introduced in Section III, where characteristics of biomedical NEs are presented. Section IV introduces the BA method constructed by a novel neural network architecture. Experiments are conducted in Section V, in which the result of each step in the BA method is presented for a better understanding of biomedical NE recognition based on our BA method. The conclusion is given in Section VI.

II. RELATED WORK
The task to recognize NEs was first proposed in the Sixth Message Understanding Conference (MUC-6) [43]. It was originally designed as a fundamental task to support more complicated tasks, e.g., relation extraction and event recognition. Unlike other tasks based on information retrieval that work at the document level, NE recognition is a typical task that extracts designated linguistic units from a sentence. Because a sentence often includes a limited number of words, the task of recognizing NEs suffers from a serious feature sparsity problem. In recent decades, significant progress has been made. The task has supported various applications and considerably accelerated the development of information extraction. However, its performance still needs improvement, especially for biomedical NE recognition, where nested and discontinuous structures are widely adopted. In this section, related work about NE recognition is given, which is helpful for understanding the development of NE recognition.

In the early stage, handcrafted rules were widely used to identify NEs. For example, Budi et al. [21] constructed rules composed of grammatical (e.g., part-of-speech), syntactic (e.g., word precedence) and orthographic patterns (e.g., capitalization). Rule-based approaches usually rely on handcrafted rules. These approaches can achieve good results when rules are carefully designed to model the underlying corpus. The problem with handcrafted rules is that domain-specific experts are required to generate rules. It is expensive in human labor and time consuming. Furthermore, it is not effective to transform established rules from one domain to another. To avoid these shortcomings, many studies focus on automatic rule generation. For example, Etzioni et al. [44] proposed a semisupervised framework that divides the process of recognizing NEs into three steps: pattern learning, subclass extraction and list extraction. It automatically generates new extraction rules and enables new entity types to be extracted. Eftimov et al. [45] combined NLP techniques with Boolean algebra rules and matrix theory to support knowledge extraction.

Machine learning-based methods can automatically learn a decision boundary from annotated data [46]. The task is modeled as a multiclassification or a sequence tagging task. It has received great success in NE recognition. In this field, many supervised algorithms have been applied, e.g., decision tree [12], the maximum entropy model [13], the support vector machine (SVM) [14], the hidden Markov model (HMM) [11] and the conditional random field (CRF) [15]. For supervised methods, extracting valid features is important to support NE recognition. Various types of features have been proposed, e.g., word-level features (e.g., morphology, part-of-speech, capital), text-level features (e.g., bag-of-word features, TF-IDF) and contextual features (e.g., syntactic or semantic information around a word). In addition to supervised methods, unsupervised methods have also been explored to support the recognition task. For example, Collins et al. [20] presented two unsupervised algorithms for NE recognition. They designed a system that can be started with just seven simple “seed” rules. They demonstrated that unsupervised methods can reduce the requirement for annotated data.

In recent years, neural networks have shown great potential to support NLP tasks. Compared to traditional machine learning approaches that are based on manually designed features, neural networks can extract high-order abstract features from raw input automatically. It is effective to stack different layers (e.g., convolutional layer, recurrent layer, pooling layer and fully connected layer) to implement complex nonlinear feature transformation.

Many neural network models have been proposed for NE recognition, e.g., CNN [47], LSTM [49], LSTM-CNNs [23], attention [48], Bi-LSTM [18], and LSTM-CRF [22]. For example, Chiu et al. [23] extracted character-level features through a convolutional layer, concatenated them with word-level features and fed them into a Bi-LSTM. It has the advantage of using character-level features such as the prefix and suffix. Because Bi-LSTM has the advantage of modeling the dependency in a sentence, it is the most typical method for NER. Huang et al. [22] systematically compared the performance of LSTM, Bi-LSTM, LSTM + CRF and Bi-LSTM-CRF. In the biomedical field, Gridach et al. [24] combined neural networks and the CRF model, which showed promising performance for biomedical NE recognition.

Many previous works have mainly focused on recognizing NEs with flattened structures. The earliest study of nested NEs was conducted by Alex et al. [8], where three traditional nested NE recognition approaches were compared: the layering approach, the cascading approach and the joint approach. Finkel et al. [26] used a parsing tree to recognize nested NEs, in which rules were designed to append entity candidates into a parsing tree. Then, a CRF model was implemented on the tree to output a normalized labeling sequence. Nam et al. [25] proposed a deep neural network to recognize
the outermost and innermost NEs. These models are mainly based on sequence models to resolve the nestification problem without considering the discontinuous NE problem.

There are models designed to recognize nested NE directly. Lu et al. [27] presented mention hypergraphs to recognize nested NEs. A hypergraph is a compact representation of all possible combinations of nested NEs. Based on this representation, a log-linear approach is implemented to label each subhypergraph for recognizing nested NEs. This model was developed by researchers such as Muis et al. [28] and Katiyar et al. [29]. Sohrab et al. [31] proposed a “deep exhaustive” model in which all possible regions (less than a fixed length) in a sentence were enumerated as NE candidates and then classified by a classifier. Wang et al. [32] presented a scalable transition-based method to model the nested structure of NEs. They map a sentence with nested mentions to a designated forest. Then, a Stack-LSTM was used to represent the states of the model in a continuous space. Ju et al. [33] identified nested entities by producing a flattened NER layer from the output of the previous LSTM layer. This model dynamically stacks the flattened NER layer until no external entities can be extracted. Lin et al. [34] proposed a sequence-to-nugget architecture that uses a head-driven phrase structure for nested NE recognition. Recently, pretrained language models have shown valuable potential to improve performance [53]–[58].

III. MATERIALS

This paper focuses on recognizing nested NEs from biomedical texts. The GENIA dataset [2] is adopted to evaluate our method to recognize nested biomedical NEs. ACE 2005 is a popular open evaluation corpus [4], in which the nested NE problem has been widely studied. Therefore, our method is also evaluated on the ACE corpus.

The GENIA dataset was created to develop and evaluate molecular biology information retrieval and text mining systems by the GENIA project. The dataset was collected from biomedical literature, which contains 2,000 abstracts in MEDLINE by PubMed based on the three medical subject heading terms: human, blood cells and transcription factors. This dataset contains 36 fine-grained entity categories. There are 94,584 NEs in total. The ACE 2005 English dataset was collected from broadcasts, newswires and weblogs. It contains 506 documents annotated with 7 types. The ACE is annotated with 41131 NEs. Statistical information about the GENIA and the ACE corpora is listed in Table I.

Discontinuous NEs often have structures such as [prefix_1, prefix_2, ..., conjunction prefix_N head]. For example, “HEL, KU812 and K562 cells” denote three discontinuous entities: “HEL cells”, “KU812 cells” and “K562 cells”. Because discontinuous NEs are difficult to model, they are totally ignored by many existing methods [27], [28], [34]. However, according to the ACE annotation guideline [4], an NE can refer to an entity or a set of entities. Therefore, we transform discontinuous NEs into nested structures. With this notation, the previous example can be transformed into three nested NEs: “HEL, KU812 and K562 cells”, “KU812 and K562 cells”, “K562 cells”. After discontinuous NEs were transformed into nested NEs, the nested NE ratio increased to 35.27% in the GENIA dataset.

In nested NEs, one property that affects the recognition performance is the number of nested layers. It determines the difficulty of recognizing nested NEs. Table II shows information about NEs in different nesting layers.

Another important issue about nested NEs is the length of the NE. It also indicates the difficulty level of recognizing nested NEs. Figure 2 shows the distribution of NEs in the GENIA dataset and the ACE dataset.
IV. METHOD

In this paper, we use the boundary assembly (BA) method to implement the biomedical NE recognition task. It is a cascaded framework focusing on nested NE recognition, which consists of three steps: boundary detecting, boundary assembling and entity discriminating. The framework of the BA method is shown in Fig. 3.

In Figure 3, the input of the BA method is a sentence. A sequence model (e.g., CRF, HMM) can be implemented to detect entity boundaries. Then, boundaries are assembled into NE candidates by assembling strategies (e.g., a greedy matching method). After entity candidates are generated, a classifier is adopted to find positive cases.

In Chen et al. [9], a CRF model is implemented to identify entity boundaries. The entity identification is implemented by a maximum entropy model. In recent work, motivated by the development of neural networks, Chen et al. [1] developed a BA method with a Bi-LSTM-CRF architecture. Because the neural network has the advantage of extracting high-order abstract features automatically and utilizing pretrained word embeddings learned from large-scale raw data, the neural network-based BA method shows impressive improvement for Chinese nested NE recognition.

In this paper, a new neural network structure is designed to detect entity boundaries that follow the characteristics of English. In the following subsections, we discuss the three steps in detail.

A. BOUNDARY DETECTING

Identifying NE boundaries is the first step of the BA method, in which the boundary detection quality has a great impact on the final performance. First, highly accurate boundary detection can collect more correct entities and increase the recall rate of entity recognition. It also reduces the ratio of false entity candidates in the boundary assembling step.

The boundary recognition task is modeled as a sequence labeling problem. If the “S-E-O” (start, end, outside) encoding is adopted, a classifier can be implemented to support the boundary detection task. However, this strategy can lead to ambiguity when an NE is represented as a single word because it should be assigned an “S” tag and an “E” tag simultaneously. To avoid this problem, we implement two classifiers, which adopt the “S-O” and “E-O” encoding.

Figure 4 shows the model to detect NE boundaries. It is a classic neural network-based sequence labeling model composed of an input layer, a lookup table, a Bi-LSTM layer, a full connection layer and an output layer. Each layer is discussed as follows.

1) INPUT LAYER

The input layer receives a sequence of words (a sentence) each time. For a neural network, category numbers (words or indexes of words) cannot be processed directly. Therefore, words in a sentence are mapped into word vectors through a lookup table.

2) EMBEDDING LAYER

This layer transforms indexes of words as continuous vectors. Two types of embeddings are used in this layer: word-level embedding and character-level embedding. Word-level embedding is generated through a lookup table. The lookup table can be initialized randomly or with pretrained values. In our work, the GloVe embeddings,^2^ BERT [38] and BioBERT [51] are adopted to initialize the lookup table. Because a pretrained lookup table is learned from external

---

^2^ A bidirectional long short-term memory with conditional random fields.

^3^ The GloVe embedding was presented by Stanford University publicly, which trained on 6 billion words from Wikipedia and web text [39].
resources, it encodes semantic information about words. The character-level embedding contains information within a word, e.g., prefix or suffix [30]. The character-level embedding can be learned by an RNN or a CNN model, referred to as char_RNN and char_CNN, respectively (Figure 4 shows the char_RNN model).

To assess the impact of word embedding and character embedding on performance, we compare two neural network structures (RNN, CNN) with three embeddings (GloVe, BERT, BioBERT), referred to as GloVe + BERT, BioBERT), referred to as GloVe + char_RNN, GloVe + char_CNN, BERT and BioBERT (BERT and BioBERT already contain character-level information). GloVe (or BERT) maps a word into a 100 (or 768) dimensional vector. char_RNN embedding and char_CNN embedding initialize uniform samples with a dimension size of 30.

3) Bi-LSTM LAYER

LSTM is a special RNN model that overcomes the gradient disappearance/exposure problem caused in a long sequence [52]. The LSTM model selectively stores context information through a specifically designed gate structure, which enables the model to memorize long dependency information. The main structure of the LSTM network can be expressed formally as:

\[
\begin{align*}
    f_t &= \sigma(W_f \cdot [h_{t-1}, x_t] + b_f) \\
    i_t &= \sigma(W_i \cdot [h_{t-1}, x_t] + b_i) \\
    C_t &= \tanh(W_c \cdot [h_{t-1}, x_t] + b_C) \\
    C_t &= f_t \odot C_{t-1} + i_t \odot C_t \\
    o_t &= \sigma(W_o \cdot [h_{t-1}, x_t] + b_o) \\
    h_t &= o_t \odot \tanh(C_t)
\end{align*}
\]

where \( \sigma \) is the elementwise sigmoid function and \( \odot \) is the Hadamard product. \( \tanh \) represents the hyperbolic tangent activation function. \( x_t \) is the input vector at time \( t \). \( i_t, f_t, \) and \( o_t \) denote the input gate, forget gate and output gate at time \( t \), respectively. \( W \) and \( b \) are the weight matrix and bias vectors. \( C_t \) represents the state at time \( t \). \( h_t \) is the output at time \( t \).

In our work, we use a bidirectional LSTM (Bi-LSTM) structure [18]. It is effective for capturing both future and past dependencies in a sentence. An LSTM outputs 500-dimensional vectors. They are concatenated and fed into a multilayer perceptron (MLP) layer, which contains three dense layers transforming the vector from 500 to 100, 20, and 2 dimensions.

4) CRF LAYER

This layer is used to obtain a globally optimized label sequence. It outputs a structuralized label sequence. Formally, let \( X = \{x_1, x_2, \ldots, x_n\} \) be an input sequence. \( P_{n \times k} \) denotes the tag distribution outputted by an MLP layer, where \( k \) is the number of tag types. In our model, \( k = 2 \) corresponds to the “B-O” encoding or “E-O” encoding. \( p_{i,j} \) is the probability that the \( i \)th word is marked as the \( j \)th tag. Given an input \( X \), the label sequence \( Y = \{y_1, y_2, \ldots, y_n\} \) can be computed as:

\[
s(X, Y) = \sum_{i=0}^{n} A_{i,j} + \sum_{i=1}^{n} p_{ij}
\]

where \( A \) is the transition matrix. \( A_{i,j} \) represents the score from the label \( i \) to label \( j \). The probability of an optimized label sequence is:

\[
p(Y|X) = \frac{e^{s(X, Y)}}{\sum_{\hat{Y} \in Y_X} e^{s(X, \hat{Y})}}
\]

where \( Y_X \) represents a collection of all possible label sequences, even if those do not verify the B-O encoding.

For a training set, the likelihood function of label sequences is:

\[
\log(p(Y|X)) = s(X, Y) - \log(\sum_{\hat{Y} \in Y_X} e^{s(X, \hat{Y})})
\]
In the training process, parameters maximizing the log-likelihood \( \log(p(Y|X)) \) are chosen. The CRF layer outputs a label sequence, which has the highest conditional probability:

\[
y^* = \arg\max_{\hat{y} \in \hat{y}_X} s(X, \hat{y})
\]

(5)

**B. BOUNDARY ASSEMBLING**

After NE boundaries were detected, they were assembled into NE candidates to support further evaluation. This is the key process for generating nested NEs. Many strategies can be employed to implement this process. For example, Chen et al. [9] presented two methods: \( n \) left (right) pairing method and \( n \) left (right) greedy matching method. The \( n \) left pairing method uses an end boundary matching top \( n \) possible start boundaries in the range between this boundary and a left end boundary. The \( n \) left greedy matching method uses end boundary greedy matching \( n \) start boundaries on the left side.

Chen et al. [9] showed that the assembling strategy has a great impact on NE recognition performance. Experiments indicate that generating more NE candidates can increase recall and hurt precision. In our work, a greedy algorithm is used to generate NE candidates, as shown in Figure 5.

In Figure 5, \( n \) left greedy matching (\( n \)-LGM) uses every end boundary greedily matching \( n \) start boundaries on the left side. In this paper, we compare the performance between \( n = 1, n = 2 \) and \( n = A \) (ALL), where “A” represents a boundary matching all boundaries on the left (or right) side of a sentence.

Some existing methods (e.g., Xu et al. [50] and Xia et al. [53]) generate every possible NE candidate with a certain length. They generate a large number of false NE candidates, which leads to a serious data imbalance problem. Additionally, the BA method to generate NE candidates takes advantage of boundary information. It helps avoid data imbalance and decrease computational complexity.

**C. ENTITY DISCRIMINATING**

The entity discriminating step focuses on identifying true NEs from entity candidates generated in the boundary assembling step. The input of this step contains sentences with marked NE candidates. Each candidate has a label to indicate whether it is a true NE. Therefore, the input can be represented as a set \( \{ (E_{[i_1,j_1]}, S_1, L_1), (E_{[i_2,j_2]}, S_2, L_2), \ldots, (E_{[i_k,j_k]}, S_n, L_m) \} \), where \( E_{[i_k,j_k]} \) (\( 1 \leq k \leq n \)) represents an NE candidate with the range from the \( i_k \)th word to the \( j_k \)th word in the sentence \( S_k \). It is marked with a label \( L_k \) denoting its entity type.

The task can be described as follows: given a sentence in which an NE candidate is labeled, a classifier predicts whether the candidate is a true NE. In this step, there are two issues to be considered. First, there might be two NE candidates (e.g., \( E_{[i_1,j_1]} \) and \( E_{[i_2,j_2]} \)) that contain the same words but are labeled with different labels because the type of NE candidate heavily depends on its context \( (S_i, S_j) \). For example, a biomedical entity “staphylococcal enterotoxin A” can be labeled as a “other_organic_compound” or a “protein_molecule”.

Second, a sentence may contain several NE candidates. Simply implementing a convolutional network on the whole sentence is not feasible to distinguish them because these candidates share the same contexts. Another approach for resolving this problem is to use features around each \( E_{[i_k,j_k]} \). However, it is not effective to capture global features in a sentence.

As discussed above, the entity discriminating task is not a simple sentence classification problem. In our work, we design a multichannel CNN model, referred to as multiCNN, to implement the entity discriminating task. The structure of this model is shown in Figure 6.

In Figure 6, the multiCNN model is composed of three parallel CNNs, an MLP layer and a softmax layer. Each layer is discussed as follows:

1) **INPUT LAYER**

   Given a marked NE candidate, the sentence is divided into three channels: left context, NE candidate, and right context.
The length of every channel is fixed as 80. Each channel is processed by a stacked neural network composed of an embedding layer, a convolutional layer and a max-pooling layer.

2) EMBEDDING LAYER
In this layer, BERT and BioBERT models are also used to generate word embedding directly. Every word is transformed into a 768-dimensional vector by a pretrained lookup table initialized with BERT and BioBERT. This layer is the same as the embedding layer in the boundary detection step.

3) CONVOLUTIONAL LAYER
The convolutional layer implements a convolution operation on a vector sequence outputted by the embedding layer. Let \([x_1, x_2, \cdots, x_n]\) (referred to as \(x_{1:n}\)) denote the output of an embedding layer. Assume that \(x_{i:i+k}\) is a subsequence of \(x_{1:n}\). Then, the convolution operation can be formalized as:

\[ C_i = f(W^T \cdot x_{i:i+k} + b) \]

where \(W \in \mathbb{R}^{K \times H}\) is the filter of the convolution operation and \(b\) is a bias vector with dimension \(H\). \(f\) denotes a nonlinear function, e.g., hyperbolic tangent. The dimension size of output \(C_i\) is \(H\).

After the convolution operation is implemented through input \(x_{1:n}\), it outputs a vector sequence \([c_1, c_2, \cdots, c_n]\), which represents abstract features depending on the input \(x_{i:i+k}\). In our work, the convolutional layer is composed of 10 convolution kernels 5, 7, 9, and 11 in width.

4) MAX-POOLING LAYER
This layer is used to collect representative features from the output of the convolutional layer. The operation can be simply represented as \(max\{c_1, c_2, \cdots, c_{n-K+1}\}\). The output of the max-pooling layer is concatenated into a flattened vector and fed into an MLP layer. The dimension size of the max-pooling layer is 512.

5) MLP LAYER
MultiCNN contains three channels. Each channel receives a raw input sequence from the input layer. After the embedding operation, the convolutional operation and the pooling operation, each channel outputs a vector representing high-order abstract features. Because every channel is implemented independently, it is not feasible to model the interaction between them. Therefore, an MLP layer is used to provide global regulation between the three channels. The MLP layer is composed of two dense layers (named the fully connected layer). They map input vectors from 512 to 100 and from 100 to 11 (class number of GENIA). Then, a softmax is used to normalize the output.

V. EXPERIMENTS
In our experiments, the GENIA corpus [2] is mainly adopted to evaluate the BA method for biomedical NE recognition. The ACE 2005 [4] is also used for comparison. The data are partitioned into training data, developing data and testing data with a ratio of 8:1:1, the same as [27], [28] and [33]. The performance is reported based on traditional precision (P), recall (R) and F1-score (F) measurements. All settings of neural networks of the BA method are given in Section IV.

The boundary recognition classifier and the entity candidate classifier use the same “AdamWeightDecay” optimizer. The learning rate, weight decay rate and batch size are set as 5e-6, 0.01 and 48, respectively. To reduce the overfitting problem, a dropout regularization with a value of 0.5 is set after the embedding layer.

This section is divided into four parts. Section V-A discusses factors influencing the performance of boundary detection. In Section V-B, the influence of the assembly strategy is presented. In Section V-C, the final performance of the BA is given. In Section V-D, our method is compared with other state-of-the-art work using GENIA data and ACE data.

A. RESULT OF BOUNDARY DETECTING
Finding entity boundaries precisely are very important to support the biomedical NE recognition task. Compare with NEs in a sentence, boundaries are linguistic units with small granularity. It depends more on local features in a sentence. Therefore, it can avoid the influence caused by other NLP processing steps (e.g., part-of-speech labeling and parsing). Therefore, it is expected to achieve higher performance. In this section, the factors influencing boundary detection are investigated. Three factors may influence the performance of boundary detection: the structure of the neural network, the pretrained word embeddings and the discontinuous problem.

Two neural network structures are compared to evaluating the character embedding: char_CNN and char_RNN. The char_CNN implements convolutional. It is effective for collecting morpheme features from input. char_RNN is effective in capturing the semantic dependency between characters. In word embedding, pretrained word embeddings are learned from external resources. They are very effective in capturing semantic information in a sentence. In this experiment, GloVe embeddings\(^4\) and BERT [38] are adopted to initialize the lookup table. In the GENIA data, BioBERT is also available for comparison, which is trained on biomedical documents [51]. In BERT and BioBERT models, character-level information in a word is automatically embedded. Therefore, the default neural network structure for BERT and BioBERT is char_RNN. Table 3 shows the influence of network structure and pretrained word embeddings.

The results show that “char_RNN” and “char_CNN” have similar performances on the GENIA and ACE datasets. Based on GloVe embedding, the performance of GENIA is higher than the performance of ACE. The reason is that, compared

\(^4\)The GloVe embedding was presented by Stanford University publicly, which trained on 6 billion words from Wikipedia and web text [39].
TABLE 3. Influence of the neural network structure and the pretrained word embeddings.

| Dataset | Model             | Start Boundary | End Boundary |
|---------|-------------------|----------------|--------------|
|         | P(%) | R(%) | F(%) | P(%) | R(%) | F(%) |
| GENIA   | GloVe+char_RNN    | 86.37 | 79.62 | 82.85 | 86.07 | 79.73 | 82.78 |
|         | GloVe+char_CNN    | 85.60 | 80.99 | 83.23 | 85.18 | 81.91 | 83.51 |
|         | BERT              | 88.21 | 85.57 | 86.87 | 88.42 | 85.77 | 87.08 |
|         | BioBERT           | 88.56 | 85.99 | 87.26 | 88.76 | 87.44 | 88.09 |
| ACE     | GloVe+char_RNN    | 85.85 | 76.92 | 81.14 | 81.26 | 79.21 | 80.22 |
|         | GloVe+char_CNN    | 83.71 | 80.21 | 81.92 | 81.04 | 79.98 | 80.51 |
|         | BERT              | 90.45 | 92.15 | 91.29 | 88.16 | 90.88 | 89.50 |

TABLE 4. Performance on all boundary of GENIA with BioBERT embedding.

| Dataset | Model             | Start Boundary | End Boundary |
|---------|-------------------|----------------|--------------|
|         | P(%) | R(%) | F(%) | P(%) | R(%) | F(%) |
| 5 type NEs | 88.17 | 85.34 | 86.73 | 88.70 | 86.65 | 87.66 |
| 5 types + Dist. NEs | 86.72 | 84.49 | 85.59 | 88.20 | 84.92 | 86.53 |
| All type NEs | 89.21 | 82.40 | 85.67 | 89.23 | 83.23 | 86.14 |
| All types + Dist. NEs | 89.01 | 82.72 | 85.75 | 89.79 | 82.85 | 86.18 |

with the ACE, GENIA has a larger number of NEs (Table 1), which enables character embedding tuning appropriately in the training process. When BERT was adopted, the performance in the ACE data improved considerably, outperforming GENIA by approximately 9%. The main reason is that GENIA contains many specialized terms that are not registered in the BERT lookup table. On the other hand, BioBERT is pretrained on biomedical texts and has better performance than BERT.

In the GENIA data, many existing methods only focus on the five NE types and totally ignore discontinuous NEs, e.g., Lu et al. [27] and Muis et al. [28]. We conducted an experiment to study the influence of discontinuous NEs and NE types. The result is shown in Table 4.

The result shows that the performance of five types of NEs is better than the performance using all types of NEs. The result is reasonable because increasing the number of class types makes the decision boundary more complex. Furthermore, the GENIA data are unbalanced in different entity types. The number of NEs in each of the five types is larger (Table 4), which has better performance. Other entity types have a small number, which worsens the performance.

One important phenomenon is that for all types of NEs when discontinuous NEs are added, the performance for both start boundary identification and end boundary identification is improved. The reason is that discarding these NEs cannot make full use of annotated data. Furthermore, because it enforces the change in annotated labels, it hurts final performance.

In the following experiments, our method is implemented on all NEs (including the discontinuous NEs) using the GENIA dataset. Because BERT and BioBERT received the best performance on the boundary detecting task, we use them as default settings to recognize NE boundaries.

B. RESULT OF BOUNDARY ASSEMBLING

After NE boundaries are detected, they are assembled into NE candidates for further processing. The method to assemble boundaries is known as the “assembling strategy”. Several assembling strategies have been discussed in Chen et al. [9]. In our work, the greedy matching strategy is adopted. In Table 5, “n-LGM” (n = 1, 2) represents the strategy that an end boundary is greedily matched to the nearest n left start boundary. “n-RGM” is the strategy in which a start boundary is greedily matched to the nearest n right end boundary. “All” means that an end boundary matches every start boundary on the left side. Because a single word can be an NE, every word with “S” and “E” labels are also collected as an NE candidate. The performance given in Table 5 is computed from the generated true and false NE candidates.

The result shows that when the value of n is increased, the precision performance is decreased. Meanwhile, the recall is increased. In previous work, Chen et al. [1] showed that a high recall ratio is important to support NE recognition, and the highest performance is received when the setting “All” is adopted.

C. RESULT OF ENTITY DISCRIMINATING

Based on NE candidates, a classifier discussed in Figure 6 is implemented to classify NE candidates. Table 6 show the performance of each NE type in the GENIA and ACE 2005 test sets.

To evaluate the BA method, two nested NE recognition methods are conducted for comparison: the cascading model and the layering model discussed in Alex et al. [8]. Both use the same data and settings as the BA boundary recognition model (BioBERT + RNN and BERT + RNN). The results are listed in Table 6.

The cascading model recognizes each NE type with an independent classifier. It is not effective to use annotated data. Furthermore, it cannot recognize nested NEs in the same NE type. In the layering model, the first two classifiers are independently implemented to recognize the innermost or outermost layer of a sentence. Then, outputs are merged to give the final performance. Compared with the cascading model, the layering model can obtain better performance when training data of an NE type is sufficient (e.g., “protein”, “other_name”, “DNA” and “Cell type”). However, the result also indicates that the layering model is very sensitive to the number of NEs. For example, the numbers of NEs “lipid”, “virus” and “multicell” are 2,362, 2,135 and 1,790, respectively. They all received very low performance. Comparing the BA method with the cascading model and layering model, the performance of the BA method is improved considerably.
The bottom of Table 6 shows the entity performance for each category in the ACE 2005 test data set. The performance of ACE is better than that of GENIA. The main reason for the lower performance in GENIA is that it contains a large number of abbreviated forms of NEs, and a hyphen is widely used to generate NEs, for example, “EGR2”, “beta-gal”, “NF-E1”, “GF-1”. They have at least two negative factors on the final performance. First, these words are rarely registered by a pretrained word embedding lookup table. They are often initialized as a random vector that cannot capture semantic information of words by external resources. Second, abbreviations are less likely (lexical features) to express their semantic information.

D. COMPARING WITH OTHER METHODS

In the GENIA corpus, researchers often report the performance on five NE types (DNA, RNA, protein, cell line and cell type). To compare with existing methods, we generate results on the five NE types with the same settings as Lu et al. [27], Muis et al. [28], and Ju et al. [33]. We compare with several recently published state-of-the-art models. They are listed as follows:
As discussed above in Section II, Lu et al. [27], Muis et al. [28] and Katiyar et al. [29] are hypergraph-based methods. Wang et al. [32] is a scalable transition-based method. Wang et al. [32] is a scalable transition-based method. Lin et al. [34] proposed a sequence-to-nugget architecture that uses a head-driven phrase structure for nested NE recognition. In Table 7, the BERT is used in Xia et al. [53], Fisher et al. [54], Shibuya et al. [55], Straková et al. [57] and Jue et al. [58]. Compared with them, our model achieves state-of-the-art performance in the task of nested NE recognition.

VI. CONCLUSION

The BA method is a cascading framework to recognize nested NEs. It divides the task into three steps: detecting boundaries, assembling candidates and discriminating true NEs. Because boundaries are linguistic units with small granularity, they have less ambiguity and can be detected with high performance. After boundaries were assembled into NE candidates, global information of a sentence can be adopted to make a final prediction. Based on the framework, two models are designed to detect NE boundaries and recognize NE candidates. In the first model, character-level features in every English word are embedded to capture morpheme information to support boundary detection. In the second model, a multiCNN model is designed to learn NE representations from separated NEs and contexts. The results showed impressive improvement.

In future work, the BA method can be extended from two aspects. The first is to improve the performance of boundary detection, which is influential on the final performance.
Because of English characteristics (such as abbreviations, hyphens), the boundary clue of English words is not as obvious as in Chinese. Making better use of multigrain features will be helpful to improve performance. The second aspect is in the discrimination step. Because a sentence usually contains several NEs, predicting an NE candidate relative to a sentence should explore structural features and semantic features in a sentence. The current multiCNN model is mainly based local features of a sentence. It still has room for improvement by capturing more global features. Furthermore, an end-to-end framework to support the boundary-based NE model is also promising, which enables global optimization.

REFERENCES

[1] Y. Chen, Y. Wu, Y. Qin, Y. Hu, Z. Wang, R. Huang, X. Cheng, and P. Chen, “Recognizing named named entity based on the neural network boundary assembling model,” IEEE Intel. Syst., vol. 35, no. 1, pp. 74–81, Jan. 2020.
[2] J.-D. Kim, T. Ohta, Y. Tateisi, and J. Tsujii, “GENIA corpus—A semantically annotated corpus for bio-textmining,” Bioinformatics, vol. 19, no. 1, pp. 1180–1182, Jul. 2003.
[3] S. Pyyalo, P. Ginter, J. Heimonen, J. Björne, J. Boberg, J. Järvinen, and T. Talakoski, “Bioinf: A corpus for information extraction in the biomedical domain.” BMC Bioinf., vol. 8, no. 1, p. 50, Dec. 2007.
[4] G. R. Doddington, A. Mitchell, M. A. Przybock, L. A. Ramshaw, S. Strassel, and R. M. Weischedel, “The automatic content extraction (ace) program-tasks, data, and evaluation.” in Proc. LREC, vol. 2, 2004, pp. 837–840.
[5] D. Benikova, C. Biemann, M. Kisselew, and S. Pdh, “GermEval 2014 named entity recognition task: Companion paper,” in Proc. GermEval Named Entity Recognit. (NER) Shared Task Workshop KONVENS, 2014. [Online]. Available: http://nbn-resolving.de/urn:nbn:de:gbv:114-rnr-2014-s-3006
[6] K. Byrne, “Named entity recognition in historical archive text,” in Proc. Int. Conf. Semantic Comput. (iSCC), Sep. 2007, pp. 589–596.
[7] L. Kelly, L. Goueriout, H. Suominen, T. Schreck, G. Leroy, D. L. Mowery, S. Velupillai, W. W. Chapman, D. Martinez, G. Zucon, and J. Palotti, “Overview of the share/clef ehealth evaluation lab,” in Proc. CLEF, Cham, Switzerland: Springer, 2014, pp. 172–191.
[8] B. Alex, B. Haddow, and C. Grover, “Recognising named entities in biomedical text,” in Proc. BioNLP, Stroudsburg, PA, USA: Association for Computational Linguistics, 2007, pp. 65–72.
[9] Y. Chen, Q. Zheng, and P. Chen, “A boundary assembling method for chinese entity-mention recognition,” IEEE Intell. Syst., vol. 30, no. 6, pp. 50–58, Nov. 2015.
[10] L. Ratinov and D. Roth, “Design challenges and misconceptions in named entity recognition,” in Proc. CoNLL, Stroudsburg, PA, USA: Association for Computational Linguistics, 2009, pp. 147–155.
[11] S. R. Eddy, “Hidden Markov models,” Current Opinion Struct. Biol., vol. 6, no. 6, pp. 361–365, 1996.
[12] J. R. Quinlan, “Induction of decision trees,” Mach. Learn., vol. 1, no. 1, pp. 81–106, 1986.
[13] J. N. Kapur, Maximum-Entropy Models in Science and Engineering. Hoboken, NJ, USA: Wiley, 1989.
[14] M. A. Hearst, S. T. Dumais, E. Osman, J. Platt, and B. Scholkopf, “Support vector machines,” IEEE Intell. Syst. Appl., vol. 13, no. 4, pp. 18–28, Jul/Aug. 2008.
[15] J. Lafferty, A. McCallum, and F. C. N. Pereira, “Conditional random fields: Probabilistic models for segmenting and labeling sequence data,” in Proc. 18th Int. Conf. Mach. Learn. (ICML), 2001, pp. 282–289. [Online]. Available: http://portal.acm.org/citation.cfm?id=655813
[16] G. Zhou and J. Su, “Named entity recognition using an HMM-based chunk tagger,” in Proc. 40th Annu. Meeting Assoc. Comput. Linguistics. Stroudsburg, PA, USA: Association for Computational Linguistics, 2002, pp. 473–480.
[17] R. Leaman and Z. Lu, “TaggerOne: Joint named entity recognition and normalization with semi-Markov models,” Bioinformatics, vol. 32, no. 18, pp. 2839–2846, Sep. 2016.
O. Etzioni, M. Cafarella, D. Downey, A.-M. Popescu, T. Shaked, S. Soderland, D. S. Weld, and A. Yates, “Unsupervised named-entity extraction from the web: An experimental study,” Artif. Intell., vol. 165, pp. 91–134, Jun. 2018.

T. Eftimov, B. K. Seljak, and P. Korosec, “A rule-based named-entity recognition method for knowledge extraction of evidence-based dietary recommendations,” PLoS ONE, vol. 12, no. 6, Jun. 2017, Art. no. e0179488.

C. M. Bishop, Pattern Recognition and Machine Learning (Information science and statistics). New York, NY, USA: Springer, 2006, p. 738.

K. Yao, G. Zweig, M.-Y. Hwang, Y. Shi, and D. Yu, “Recurrent neural networks for language understanding,” in Proc. INTERSPEECH, 2013, pp. 2524–2528.

Y. Cui, Z. Chen, S. Wei, S. Wang, T. Liu, and G. Hu, “Attention-over-attention neural networks for reading comprehension,” 2016, arXiv:1607.04423. [Online]. Available: http://arxiv.org/abs/1607.04423

S. Ravuri and A. Stolcke, “Recursive neural network and LSTM models for lexical utterance classification,” in Proc. INTERSPEECH, 2015, pp. 2524–2528.

M. Xu and H. Jiang, “A FOFE-based local detection approach for named entity recognition and mention detection,” 2016, arXiv:1611.00801. [Online]. Available: http://arxiv.org/abs/1611.00801

J. Lee, W. Yoon, S. Kim, D. Kim, S. Kim, C. H. So, and J. Kang, “BioBERT: Pre-trained biomedical language representation model for biomedical text mining,” 2019, arXiv:1901.0874. [Online]. Available: https://arxiv.org/abs/1901.0874

F. A. Gers, J. Schmidhuber, and F. Cammins, “Learning to forget: Continual prediction with LSTM,” in Proc. 9th Int. Conf. Artif. Neural Netw., 1999, pp. 850–855.

C. Xia, C. Zhang, T. Yang, Y. Li, N. Du, X. Wu, W. Fan, F. Ma, and P. Yu, “Multi-grained named entity recognition,” 2019, arXiv:1906.08449. [Online]. Available: http://arxiv.org/abs/1906.08449

J. Fisher and A. Vlachos, “Merge and label: A novel neural network architecture for nested NER,” 2019, arXiv:1907.00464. [Online]. Available: http://arxiv.org/abs/1907.00464

T. Shibuya and E. Hovy, “Nested named entity recognition via second-best sequence learning and decoding,” 2019, arXiv:1909.02250. [Online]. Available: http://arxiv.org/abs/1909.02250

Y. Luan, D. Wadden, L. He, A. Shah, M. Ostendorf, and H. Hajishirzi, “A general framework for information extraction using dynamic span graphs,” 2019, arXiv:1904.03296. [Online]. Available: http://arxiv.org/abs/1904.03296

J. Straková, M. Straka, and J. Hajic, “Neural architectures for nested NER through linearization,” 2019, arXiv:1908.06926. [Online]. Available: http://arxiv.org/abs/1908.06926

J. Wang, L. Shou, K. Chen, and G. Chen, “Pyramid: A layered model for nested named entity recognition,” in Proc. 58th Annua. Meeting Assoc. Comput. Linguistics, 2020, pp. 5918–5928.

YANPING CHEN is currently an Associate Professor with the College of Computer Science and Technology, Guizhou University, Guiyang. His research interests include artificial intelligence and natural language processing.

YING HU is currently pursuing the degree with the College of Computer Science and Technology, Guizhou University, Guiyang. Her research interest includes natural language processing.

YIJING LI is currently pursuing the degree with the College of Computer Science and Technology, Guizhou University, Guiyang. Her research interest includes natural language processing.

RUIZHANG HUANG is currently an Associate Professor with the College of Computer Science and Technology, Guizhou University, Guiyang. Her research interests include information retrieval and text mining.

YONGBIN QIN is currently a Professor with the College of Computer Science and Technology, Guizhou University, Guiyang. His research interests include big data processing, cloud computing, and text mining.

YUEFEI WU is currently pursuing the degree with the College of Computer Science and Technology, Guizhou University, Guiyang. His research interest includes natural language processing.

QINGHUA ZHENG (Member, IEEE) is currently a Professor with the Department of Computer Science and Technology, Xi’an Jiaotong University. His research interests include multimedia distance education and computer network security.

PING CHEN (Member, IEEE) received the Ph.D. degree in information technology from George Mason University. He is currently an Associate Professor of computer science and the Director of the Artificial Intelligence Laboratory, University of Massachusetts Boston, Boston. His research interests include data mining and computational semantics.