HIT: Nested Named Entity Recognition via Head-Tail Pair and Token Interaction

Yu Wang[1], **Yun Li**[1]*, **Hanghang Tong**[2], **Ziye Zhu**[1]
[1]Jiangsu Key Laboratory of Big Data Security and Intelligent Processing
Nanjing University of Posts and Telecommunications, Nanjing, China
[2]Department of Computer Science, University of Illinois at Urbana-Champaign, Urbana, IL
{2017070114, liyun, 2016070251}@njupt.edu.cn, htong@illinois.edu

Abstract

Named Entity Recognition (NER) is a fundamental task in natural language processing. In order to identify entities with nested structure, many sophisticated methods have been recently developed based on either the traditional sequence labeling approaches or directed hypergraph structures. Despite being successful, these methods often fall short in striking a good balance between the expression power for nested structure and the model complexity. To address this issue, we present a novel nested NER model named HIT. Our proposed HIT model leverages two key properties pertaining to the (nested) named entity, including (1) explicit boundary tokens and (2) tight internal connection between tokens within the boundary. Specifically, we design (1) Head-Tail Detector based on the multi-head self-attention mechanism and bi-affine classifier to detect boundary tokens, and (2) Token Interaction Tagger based on traditional sequence labeling approaches to characterize the internal token connection within the boundary. Experiments on three public NER datasets demonstrate that the proposed HIT achieves state-of-the-art performance.

1 Introduction

Named Entity Recognition (NER) is a fundamental task in natural language processing due to the fact that the named entities often convey the key information of the text (Lample et al., 2016). It is common in many practical scenarios that the named entities have a nested structure (Finkel and Manning, 2009; Silla and Freitas, 2011). That is, an entity could contain other entities or be a part of other entities. As shown in Figure 1, the entity “the western Canadian province of British Columbia” in the first example contains two inner entities, i.e., “western Canadian” and “British Columbia”. Traditional methods often treat the NER task as a sequence labeling problem (Lin et al., 2018) and thus are primarily designed to recognize flat entities in the input sentences (as shown in the second example in Figure 1). Due to the nature of the nested entity, a token might belong to different entities. It is difficult to represent such nested structures using a single label accurately. Therefore, the performance of traditional NER methods will dramatically suffer when recognizing nested entities (Katiyar and Cardie, 2018).

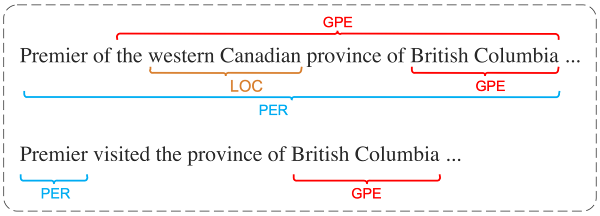
GPE
Premier of the western Canadian province of British Columbia ...
LOC
GPE
PER
Premier visited the province of British Columbia ...
PER
GPE

Figure 1: Examples of the named entity. The first example is a sentence with nested named entities, and the second one is a sentence only with flat named entities.

In recent years, more sophisticated methods have been developed for the nested NER task, which are grouped into two categories, including sequence-based method and hypergraph-based method. The sequence-based methods (Sohrab and Miwa, 2018; Ju et al., 2018; Zheng et al., 2019) often utilize the traditional sequence labeling approaches to learn the nested structure. For example, Ju et al., (2018) leverage the hierarchical Long Short Term Memory (LSTM) networks to capture the nested named entities from the inner entity to the outer entity. However, such methods might still suffer from error propagation due to the fundamental limitation of sequence labeling approaches in representing the nested structure. In response, hypergraph-based methods (Lu and Roth, 2015; Wang and Lu, 2018) introduce the hypergraph structure for learning the nested named entity. These methods replace the undirected graph structure, commonly used in the

Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing, pages 6027–6036,
November 16–20, 2020. ©2020 Association for Computational Linguistics

flat NER task, by the directed hypergraph structure. The advantage lies in that hyperedges can naturally express the nested structure. One issue of their method (Lu and Roth, 2015) is the spurious structure of hypergraphs. Wang and Lu (2018) further propose the neural segmental hypergraphs to address this issue. However, if the input sentence is too long or there exist many entity categories, their hypergraph structure becomes too complicated, which in turn makes the optimization of such models very difficult, if not impossible.

This paper further explores the precise expression of the nested structure with appropriate model complexity to overcome these shortcomings effectively. We observe two key properties pertaining to the named entity, including (1) explicit boundary tokens and (2) tight internal connection between tokens within the boundary. For example, in Figure 1, "Premier" and "Columbia" (in the first example) are explicit boundary tokens, and the tokens within the boundary are closely connected with each other. On the other hand, although the candidate region "Premier visited province of British Columbia" (in the second example) shared the same boundary tokens "Premier" and "Columbia", the tokens within the boundary suggest this region should not be an entity. This indicates that different internal tokens greatly influence whether the region determined by the boundary tokens is a valid entity. In other words, in the NER task, one region should be identified as a named entity if it meets these two properties. More importantly, these properties are sensitive to the entities with the nested structure.

Armed with these observations, we propose a novel neural model named HIT for recognizing the named entities with the nested structure. Our proposed model effectively identifies nested named entities by modeling both the boundary tokens (referred to as "head-tail pair" in this paper) and connection relationship between tokens within the boundary (referred to as "token interaction" in this paper). To be specific, we design a head-tail detector based on the multi-head self-attention mechanism (Vaswani et al., 2017) and the bi-affine classifier (Dozat and Manning, 2016) to detect explicit boundary tokens. The main advantage of the multi-head self-attention mechanism is that it can directly learn the connection between tokens without having to consider token ordering information. Particularly, we adopt Focal Loss (Lin et al., 2017) to address the class imbalance problem in the training process. This is because the head-tail detector aims to detect all candidates of head-tail pairs, only a few of which correspond to valid entities. In addition, we design a token interaction tagger based on traditional sequence labeling approaches (Lample et al., 2016; Shang et al., 2018) to characterize the internal connection between tokens within the boundary through context. Another advantage of the token interaction tagger is that the captured internal connection features contain abundant lexical and semantic information, which can be used to predict the category of entities. By integrating the head-tail detector and token interaction tagger, we apply the region classifier to predict the entity categories. Extensive experiments on three public NER datasets, including GENIA (English) (Kim et al., 2003), GermEval 2014 (German) (Benikova et al., 2014), and JNLPBA (English) (Kim et al., 2004), reveal that our proposed HIT achieves state-of-the-art performance.

The main contributions of this paper are as follows,

- We demonstrate that the head-tail pair can effectively and precisely express the boundary information of entities with nested structure.
- We utilize token interaction tagger to characterize the internal connection between tokens within the boundary, where we reveal that token interaction has a great impact on identifying entities.
- We complete entity classification with head-tail pair and token interaction sequence while introducing a multi-task loss to train our model simultaneously.

The rest of the paper is organized as follows. Section 2 describes the details of our model. Experimental results are reported in Section 3. Section 4 reviews the related work. Section 5 concludes the paper.

2 Model

In this section, we present the HIT model in detail. Figure 2 depicts the overall architecture of our model. The HIT contains three main components, including the head-tail detector, token interaction tagger, and region classifier. For each given sentence $x = \{w_1, w_2, ..., w_m\}$, where m is the length of the sentence, HIT firstly maps the sentence x to a token representation sequence $\mathbf{x} = \{\mathbf{w}_1, \mathbf{w}_2, ..., \mathbf{w}_m\}$. The representation sequence $\mathbf{x}$ is then fed into the head-tail detector to predict whether

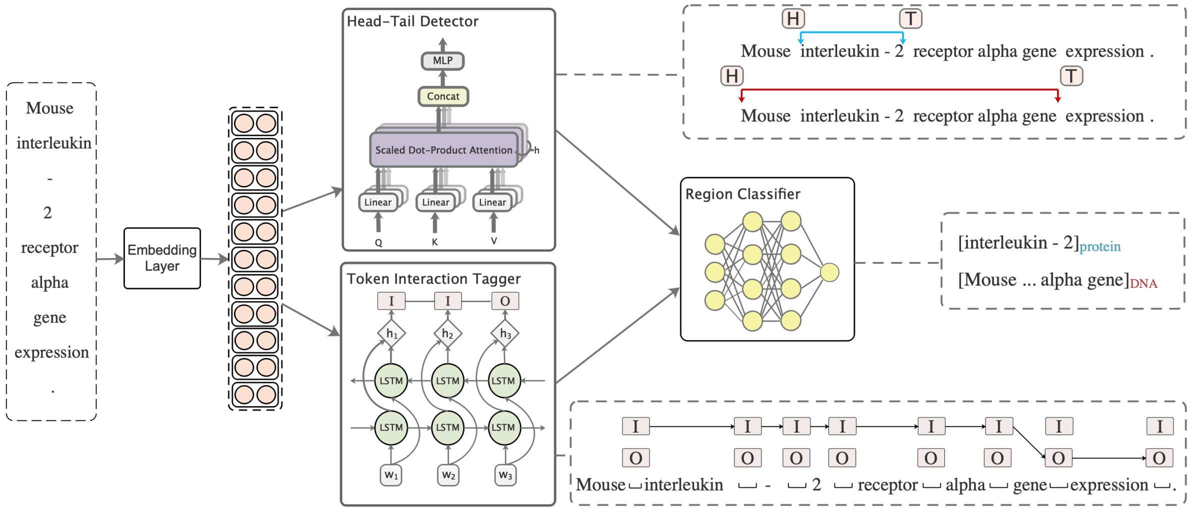

Figure 2: An overview of the proposed HIT model.

each pair-wise tokens is the head-tail of an entity. In the meanwhile, token interaction tagger is used to capture internal connections between adjacent tokens based on context, which indicates if the token before or after the current token belongs to an entity. Finally, the region classifier is employed to integrate the head-tail detector and token interaction tagger to complete the entity recognition. In the following subsections, we will describe each part of our proposed HIT in detail.

2.1 Head-Tail Detector

The head-tail detector is a pair-wise classifier that determines whether each pair of tokens in the sentence is the boundary of an entity. As shown in Figure 2, the "interleukin - 2" and "Mouse interleukin - 2 receptor alpha gene" are both entities. Ideally, our head-tail detector should be able to determine that the head-tail pairs "interleukin-2" and "Mouse-gene" are both boundary tokens of entities.

Formally, given the token representation sequence $\mathbf{x}$, the head-tail detector first generates the boundary representation $\mathbf{b}_i$ of token w_i based on the multi-head self-attention network (Vaswani et al., 2017). For simplicity, we denote the scaled dot-product attention as the following equation,

$$\text{Attention}(\mathbf{Q}, \mathbf{K}, \mathbf{V}) = \text{softmax}(\frac{\mathbf{QK}^T}{\sqrt{d_k}})\mathbf{V}, \quad (1)$$

where $\mathbf{Q}$, $\mathbf{K}$, $\mathbf{V}$ are the query matrix, keys matrix, and value matrix, respectively. In our setting, $\mathbf{Q}$ = $\mathbf{K}$ = $\mathbf{V}$ = $\mathbf{x}$, and $1/\sqrt{d_k}$ is the scaling factor. The multi-head attention can learn multiple scaled dot-product attentions by using different linear projections in parallel. Formally, the multi-head attention can be expressed as follows,

$$\begin{aligned} \text{head}_i &= \text{Attention}(\mathbf{QW}_i^Q, \mathbf{KW}_i^K, \mathbf{VW}_i^V), && (2) \\ \mathbf{b} &= \text{Concat}(\text{head}_1, \ldots, \text{head}_h)\mathbf{W}^O, && (3) \end{aligned}$$

where $\mathbf{W}_i^Q$, $\mathbf{W}_i^K$, $\mathbf{W}_i^V$, and $\mathbf{W}^O$ are trainable projection parameters.

By virtue of the self-attention mechanism, the boundary representation $\mathbf{b}_i$, composed of all the token representations, is immune from the order of tokens in the sentence. In our model, the head-tail detector is designed to detect each pair of tokens in terms of whether it is the head-tail pair of an entity, while filtering out the influence of the distance between two tokens in the sentence. Thus the self-attention mechanism is more suitable for head-tail detector than other architectures, e.g., LSTM (Lample et al., 2016) and Convolutional Neural Network (CNN) (Chiu and Nichols, 2016). It is worth pointing out that we additionally leverage the token interaction tagger (Subsection 2.2) to characterize the internal connection from the context, which takes into account the token order information.

By the generated boundary representation sequence $\mathbf{b} = \{\mathbf{b}_1, \mathbf{b}_2, ..., \mathbf{b}_m\}$, we construct token representation pairs ($\mathbf{b}_i$, $\mathbf{b}_j$) through pairwise combination, where ($\mathbf{b}_i$, $\mathbf{b}_j$) denotes that the token w_i is assumed as the head token of an entity, and w_j is assumed as the tail token. Each token representation pair is finally fed into a bi-affine classifier (Dozat and Manning, 2016) to determine whether it is the head-tail pair of an entity. The predicted head-tail distribution is defined as follow,

$$\mathbf{d}_{ij} = \mathbf{b}_i^\top \mathbf{U}^{(1)} \mathbf{b}_j + (\mathbf{b}_i \oplus \mathbf{b}_j)^\top U^{(2)} + b, \quad (4)$$

where $\oplus$ denotes concatenation operation, $\mathbf{U}^{(1)}$

and $U^{(2)}$ denote weight matrices, and b denotes bias.

In practice, the classifier does not need to consider all token representation pairs due to (Wang and Lu, 2018), which finds that restricting the maximum length of entities to 6 can cover more than 95% of entities. We set the same entity maximum length restriction (6) in our model. In addition, since only a few candidates are the boundaries of valid entities, the head-tail detector might encounter the class imbalance problem during the training process. Accordingly, we employ the Focal Loss (Lin et al., 2017) to optimize the parameters of the head-tail detector,

$$\begin{aligned} \mathcal{L}_{ht} &= \sum_{ij} - \beta'_{ij}(1-\mathbf{d}'_{ij})^{\gamma}\log(\mathbf{d}'_{ij}), \\ (\mathbf{d}'_{ij}, \beta'_{ij}) &= \begin{cases} (\mathbf{d}_{ij}, \beta_{ij}), & \text{if } \mathbf{d}_{ij} \text{ is true;} \\ (1-\mathbf{d}_{ij}, 1-\beta_{ij}), & \text{otherwise,} \end{cases} \end{aligned} \tag{5}$$

where $(1-\mathbf{d}'_{ij})^{\gamma}$ denotes the modulating factor and γ is the focusing parameter. β_{ij} denotes the weighting factor.

Note that since different entities do not share the same head-tail pair, our head-tail detector can naturally solve the difficulty of expressing nested entities. Moreover, we preserve all the predicted head-tail pairs of each sentence, which are also important features for the subsequent region classification.

2.2 Token Interaction Tagger

Although the head-tail pair is important for recognizing the nested named entity, it still ignores the connection between tokens within the head-tail pair. Inspired by (Muis and Lu, 2017) and (Shang et al., 2018), we construct a token interaction tagger to label the gap between every two adjacent tokens in the sentence. First of all, we define two possible connections of the gap, including the internal connection (I) and others (O). As shown in Figure 2, we use the internal connection (I) to indicate that both of the two adjacent tokens might belong to the same entity. The others (O) means that at least one of these two adjacent tokens do not belong to the same entity.

It is worth mentioning that we encourage the token interaction tagger to label the nested boundary gaps as the internal connection (I) when dealing with the entities with nested structure. Take an example in Figure 2 to illustrate this point, the gap between "2" and "receptor" belongs to the nested boundary gap, because the gap is inside the outer entity "mouse interleukin-2 receptor alpha gene". Such nested boundary gaps should be labeled as "I", and the explicit distinction between outer and inner entities is obtained by the head-tail detector. Therefore, the token interaction tagger is designed to capture the internal connection between adjacent tokens primarily.

Since it is important to learn lexical and semantic information in the context for determining token interaction, we employ BiLSTM to encode the token representation sequence $\mathbf{x}$. For simplicity, we denote the interaction representation extraction as the following equations,

$$\overrightarrow{\mathbf{h}}_i = \mathbf{LSTM}_f(\mathbf{w}_i, \overrightarrow{\mathbf{h}}_{i-1}, \theta_f), \tag{6}$$

$$\overleftarrow{\mathbf{h}}_i = \mathbf{LSTM}_b(\mathbf{w}_i, \overleftarrow{\mathbf{h}}_{i-1}, \theta_b), \tag{7}$$

$$\mathbf{h}_i = \overrightarrow{\mathbf{h}}_i \oplus \overleftarrow{\mathbf{h}}_i, \tag{8}$$

where the θ_f and θ_b denote the parameters of the forward and backward LSTM, respectively. The $\overrightarrow{\mathbf{h}}_i$ and $\overleftarrow{\mathbf{h}}_i$ are the hidden states at the position i of the forward and backward LSTM, respectively.

The interaction representation sequence $\mathbf{h}$ is then fed into a CRF (Lafferty et al., 2001), which can decode these features and tag connections for each gap. The scoring equation defined by CRF is

$$\begin{aligned} s(\mathbf{h}, y) = \sum_{i=1}^{m} & \log \psi_{EMIT}(y_i \to \mathbf{h}_i) \\ & + \log \psi_{TRANS}(y_{i-1} \to y_i), \end{aligned} \tag{9}$$

where y is the target tag sequence corresponding to sentence x. The $\psi_{EMIT}(y_i \to \mathbf{h}_i)$ represents the emission potential from the token w_i to the tag y_i. The $\psi_{TRANS} \in \mathbb{R}^M$ is a transition matrix that comes from CRF to control the transition probability from y_{i-1} to y_i, where M is the tag size.

We use the following loss function to optimize the parameters of token interaction tagger,

$$\begin{aligned} \mathcal{L}_i &= -\log(p(y|\mathbf{h})) \\ &= -\log(\frac{\exp(s(\mathbf{h}, y))}{\sum_{y' \in Y} \exp(s(\mathbf{h}, y'))}), \end{aligned} \tag{10}$$

where y' is one of the candidate tag sequences in Y. Since lexical and semantic information is beneficial to predict the entity categories, we retain the entire token interaction sequence $\mathbf{h}$ for the region classifier, which will be introduced in the next subsection.

2.3 Region Classifier

With the guidance of the head-tail pairs and token interaction sequence obtained from the above two components, we can establish candidate region representations. Moreover, each candidate region representation should meet the two constraints, including (1) the head-tail pair has been detected by the head-tail detector and (2) the corresponding internal tokens are closely connected (i.e., all of the token gaps within the head-tail pair labeled as internal connection (I)). Therefore, if all of the token gaps corresponding to the detected head-tail pair ($\mathbf{b}_i$, $\mathbf{b}_j$) are labeled as the internal connection (I), then we obtain the final region representation $\mathbf{r}_{ij}$ as follows,

$$\mathbf{r}_{ij} = \mathbf{b}_i^h \oplus \mathbf{b}_j^t \oplus \mathbf{c}_{ij}, \tag{11}$$

$$\mathbf{c}_{ij} = \left[\tfrac{1}{j-i}\textstyle\sum_{k=i}^{j} \mathbf{h}_k\right], \tag{12}$$

where $\mathbf{c}_{ij}$ denotes the representation of candidate token interaction, and we average the corresponding token interaction subsequence to treat them equally. The final regional representation $\mathbf{r}_{ij}$ will be sent to a two-layer multilayer perceptron networks (MLP) to predict entity category label. We compute the loss of category label prediction as follows,

$$\mathbf{d}_{ij}^r = \text{softmax}(\text{MLP}(\mathbf{r}_{ij})), \tag{13}$$

$$\mathcal{L}_r = -\textstyle\sum_{ij}(\hat{\mathbf{d}}_{ij}^r)\log(\mathbf{d}_{ij}^r), \tag{14}$$

where $\hat{\mathbf{d}}_{ij}^r$ and $\mathbf{d}_{ij}^r$ denote the true and predicted category distributions, respectively.

2.4 Training

We define the final multi-task loss as follow,

$$\mathcal{L} = \lambda_1 \mathcal{L}_{ht} + \lambda_2 \mathcal{L}_i + \lambda_3 \mathcal{L}_r, \tag{15}$$

where λ_1, λ_2, and λ_3 are hyper-parameters of $\mathcal{L}_{ht}$ in Eq. (5), $\mathcal{L}_i$ in Eq. (10), and $\mathcal{L}_r$ in Eq. (14), respectively. Note that the proposed HIT predicts the category label after all the head-tail pairs and the token interactions have been recognized. We feed all the ground-truth labels during training progress so that all components can be trained jointly. All models are optimized using the Adaptive Moment Estimation (Adam) (Kingma and Ba, 2014) method.

3 Experiments

In this section, we first introduce the datasets, the baseline methods, and implementation details. We then present the experimental results used for evaluations, followed by analyzing two key properties and the ablation study of our HIT model.

Item	Train	Dev	Test	Overall	Nested
Document	1599	189	212	2000	-
Sentences	15023	1669	1854	18546	-
Percentage	81%	9%	10%	100%	-
DNA	7650	1026	1257	9933	1744
RNA	692	132	109	933	407
Protein	28728	2303	3066	34097	1902
Cell Line	3027	325	438	3790	347
Cell Type	5832	551	604	6987	389
Overall	45929	4337	5474	55740	4789

Table 1: Statistics of GENIA dataset.

3.1 Datasets

To evaluate our proposed model, we conduct experiments on three public datasets, including GENIA (Kim et al., 2003), GermEval 2014 (Benikova et al., 2014), and JNLPBA (Kim et al., 2004). Among them, both GENIA and GermEval 2014 are commonly used benchmark datasets for nested NER task.

GENIA[1] dataset is English biology nested named entity dataset, which is based on GENIAcorpus3.02p that comes with POS tags for each token. It contains five entity types, including DNA, RNA, protein, cell line, and cell type categories. The dataset contains 18,546 sentences corresponding to 55,740 tokens. Following previous works (Finkel and Manning, 2009; Lu and Roth, 2015), we split the dataset into 8.1:0.9:1 for training, development, and testing. Table 1 shows the statistics of GENIA dataset.

GermEval 2014[2] dataset is a new German nested named entity dataset that contains four entity types. The dataset covers over 31,000 sentences corresponding to over 590,000 tokens. We use this dataset to evaluate the performance of our model in different languages.

JNLPBA[3] dataset is originally from GENIA corpus. It defines a training set and a testing set. Unlike the other two datasets, only the flat top-most entities are present in this dataset. Therefore, we use it to evaluate how well the HIT model performs in recognizing flat entities.

3.2 Baseline Methods

We compare our model with several state-of-the-art models that can be divided into two groups:

[1] http://www.geniaproject.org/genia-corpus

[2] https://sites.google.com/site/germeval2014ner/data

[3] http://www.geniaproject.org/shared-tasks/bionlp-jnlpba-shared-task-2004

Sequence-based methods. Muis and Lu (2017) label the gap between tokens by the entity separators, which can capture entities that overlap with one another. Sohrab and Miwa (2018) use the region representation by LSTM to recognize nested entities. Ju et al. (2018) encode sentence with stacking flat LSTM layers and decoding it to different categories by cascaded CRFs. Zheng et al. (2019) use the sequence labeling models to detect the nested entity boundary and merge the corresponding boundary label sequence to complete categorical prediction.

Hypergraph-based methods. Lu and Roth (2015) are the first to use the hypergraph-based method to tackle the problem of entity detection. Katiyar and Cardie (2018) learn the hypergraph representation for nested entities from the multi-layer BiLSTMs. Wang and Lu (2018) use segmental hypergraph representation to capture features and interactions that cannot be captured by previous models for nested entity recognition.

3.3 Implementation Details

For the embedding method, we initialize token vectors with 128-dimension pre-trained token embeddings, which are fine-tuned during training. We conduct hyper-parameter optimization by exploring the range of parameters shown in Table 2 using random search, and we select the set of parameters that achieves the best performance on the GENIA development set. The self-attention in the head-tail detector has a depth of 4 and heads of 4. The BiLSTM in the token interaction tagger has a depth of 2 and a hidden size of 256. The MLP in the region classifier has a depth of 2 and a hidden size of 256. The focusing parameter γ is set to 2, and the β_{ij} is set to 0.7. Moreover, the λ_1, λ_2 and λ_3 are set to 0.4, 0.3 and 0.3, respectively. The initial learning rate is set to 0.008 and decreases as the training step increases. We apply Dropout (Srivastava et al., 2014) to the output of the BiLSTM layer at the rates of 0.5. The batch size is set to 64 at the sentence level. We monitor the training process on the development set and report the final result on the test set. We implement our model under PyTorch[4]. All of our experiments are performed on NVIDIA 1080ti GPU and Intel i7-8700K CPU. The training time for each epoch is 40 min. From the performance on the development set, our model reached the best performance after 20 epochs.

Hyper-parameters	**Range**	**Final**
Self Attention–depth	[1, 4]	4
Self Attention–head	[1, 6]	4
BiLSTM–depth	[1, 4]	2
BiLSTM–hidden size	[128, 512]	256
MLP–depth	[1, 2]	2
MLP–hidden size	[128, 512]	256
Dropout	[0.2, 0.8]	0.5
γ	[0, 5]	2
β_{ij}	[0, 1]	0.7
Batch Size	[16, 64]	64

Table 2: Hyper-parameters used for training the HIT model.

3.4 Main Results

We employ the precision (P), recall (R), and F1-score (F) to evaluate the performance of each method. The experimental results of our HIT on the GENIA dataset are illustrated in Table 3. As we can see, the proposed HIT outperforms all the compared methods in both recall and F1-score, with better or comparable results in precision. For example, our HIT achieves 74.4% recall value, which surpasses Zheng et al. (2019) by 0.8%. From Table 3, we observe that all hypergraph-based methods fall short in the recall value. These results demonstrate that most entities recognized by our HIT are indeed valid entities. The reason is that the region classifier in our HIT can capture the non-entity type for the candidate region, which means that the classifier has the ability to determine whether the candidate region is a valid entity or not. With this ability of region classifier and the two constraints introduced in Section 2.3, our HIT effectively alleviates the error propagation problem. Furthermore, the HIT yields a precision value of 78.1%, which is 1.7% lower than Katiyar and Cardie (2018). On the other hand, the HIT outperforms Katiyar and Cardie (2018) by 6.2% in the recall value. More importantly, HIT outperforms Wang and Lu (2018) by 1.1%, Zheng et al. (2019) by 2.5%, and Katiyar and Cardie (2018) by 2.6% in terms of F1-score, respectively. These results indicate that our HIT is capable of capturing the explicit boundary tokens and the tight internal connection between tokens within the boundary, which precisely captures the nested structure of entities. Specifically, Table 4 shows the performance of each category on GENIA. We observe that the proposed HIT achieves

[4] https://pytorch.org/

Model	P(%)	R(%)	F(%)
(Muis and Lu, 2017)	75.4	66.8	70.8
(Ju et al., 2018)	76.1	66.8	71.1
(Sohrab and Miwa, 2018)	73.3	68.3	70.7
(Zheng et al., 2019)	75.9	73.6	74.7
(Lu and Roth, 2015)	72.5	65.2	68.7
(Katiyar and Cardie, 2018)	**79.8**	68.2	73.6
(Wang and Lu, 2018)	77.0	73.3	75.1
Our HIT	78.1	**74.4***	**76.2***

Table 3: Main results on GENIA. Significant improvement over baselines is marked with * (p-value < 0.05).

Category	P(%)	R(%)	F(%)
DNA	75.6	72.3	73.9
RNA	87.5	82.8	85.1
protein	79.4	75.6	77.5
cell line	78.2	74.3	76.2
cell type	74.9	71.2	73.0

Table 4: Results of entities for each category on GENIA test dataset.

Model	P(%)	R(%)	F(%)
(Sohrab and Miwa, 2018)	**75.0**	60.8	67.2
(Ju et al., 2018)	72.9	61.5	66.7
(Zheng et al., 2019)	74.5	69.1	71.7
Our HIT	74.8	**70.5***	**72.6***

Table 5: Main results on GermEval 2014. Significant improvement over baselines is marked with * (p-value < 0.05).

the best performance in recognizing the entities of the RNA category. The reason for the best results obtained for RNA is that the entities pertaining to RNA mainly end up either with “mRNA” or “RNA”. And our HIT yields 77.5% F1-score on the protein category, which covers over half of the named entities in GENIA.

In addition, to evaluate the performance of our proposed HIT in different languages, we conduct additional experiments on the GermEval 2014 dataset, and the experimental results are shown in Table 5. We can first observe that the HIT outperforms all the compared methods both in recall and F1-score. Compare to the suboptimal (Zheng et al., 2019), it still significantly achieves 1.4% and 0.9% relative improvements on recall and F1-score, respectively. Also, compared with Table 3, we found that the overall performance on the GENIA dataset is better than on the GermEval 2014 dataset. One possible reason is that the entities in the GermEval 2014 dataset are much sparser.

Furthermore, we conduct experiments on the JNLPBA dataset to demonstrate the applicability of our proposed HIT on flat entities. Compared with the state-of-the-art method (Gridach, 2017), which achieves 75.8% in F1-score, HIT achieves a competitive performance of 74.9%.

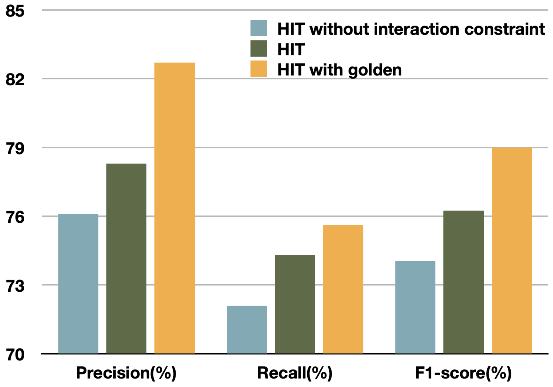

Figure 3: Results on HIT of different structures.

3.5 Analysis of Two Key Properties

Our proposed HIT is designed by leveraging two key properties pertaining to the (nested) named entity, including (1) explicit boundary tokens and (2) tight internal token connection within the boundary. In order to further evaluate the importance of these properties for nested NER, we construct the following two sets of comparative experiments on the GENIA dataset, and the corresponding experimental results are shown in Figure 3.

Analysis of Boundary Tokens. In our model, we use the head-tail pair to represent the boundary tokens of nested entities. To illustrate the importance of capturing entity boundary information in identifying nested entities, we use golden head-tail pairs instead of the results from the head-tail detector to our HIT in this set of experiments[5]. This revised model is denoted as “HIT with golden”, and the golden head-tail pairs are collected from the GENIA dataset. From Figure 3, we can find that HIT with golden achieves additional performance improvement over the proposed HIT in terms of all metrics. These results further corroborate that explicit boundary tokens indeed play an important role in recognizing named entities, and the head-tail pair can effectively and precisely express the boundary of entities with the nested structure.

Analysis of Token Interaction. In order to fur-

[5] We do not conduct opposite experiments using golden token interaction tags since our model exploits the token interaction representation to improve the overall performance, rather than the token interaction tag. In addition, it is hard to obtain the golden token interaction representation.

ther explore the effects of token interaction within the boundary, we modify the strategy of generating the candidate region representations in this set of experiments. As we introduced in Section 2.3, the candidate regions are generated under two constraints. We remove the token interaction constraint (i.e., the second constraint), which indicates the candidate region representation is only generated under the detected head-tail pairs (i.e., the first constraint). In other words, all detected head-tail pairs can establish their candidate region representations based on Eq. (11). This means that some adjacent tokens might not be closely connected together in such candidate regions. The revised model denotes as "HIT without interaction constraint". From the results shown in Figure 3, we can see that our HIT outperforms the HIT without interaction constraint by 2.4% on F1-score. The main reason is that the token interaction constraint can mitigate the error propagation caused by the head-tail detector. These results validate that the internal tokens of entity are indeed closely connected with each other, and the token interaction has a great impact on detecting named entities.

3.6 Ablation Study

We choose the GENIA dataset to conduct several ablation experiments to elucidate the main components of our proposed HIT, and the experimental results are shown in Table 6 and Table 7.

Effectiveness of Head-Tail Detector. The head-tail detector in our model consists of a multi-head attention encoder and a bi-affine classifier. To explore the effectiveness of the detector, we examine the head-tail detector based on different structures, including the BiLSTM encoder and linear classifier. In addition, in this set of experiments, we also use the Cross Entropy instead of Focal Loss to the detector. Table 6 shows the results of various head-tail detection methods. From the results, one could observe that the BiLSTM performs worse than the multi-head attention mechanism in this case. One explanation could be that the BiLSTM network learns the token ordering features and considers the distance of the head token and tail token in the sentence, which makes the BiLSTM-based detector suffer from detecting long named entities. Furthermore, we can observe that Focal Loss is more effective for the detector than Cross Entropy, due to the fact that the detector using Cross Entropy overlooks the class imbalance problem. These results validate that the Focal Loss can perform well in NLP tasks. In addition, the detector based on the bi-affine classifier achieves 1.2% improvement on F1-score compared to the detector based on the linear classifier.

Method	P(%)	R(%)	F(%)
HTD with BiLSTM encoder	79.7	77.1	78.4
HTD with linear classifier	81.2	78.9	80.0
HTD with Cross Entropy	79.2	76.7	77.9
Head-Tail Detector	82.1	80.4	81.2

Table 6: Performance of the head-tail pair detection based on the Head-Tail Detector (HTD) of different structures.

Method	P(%)	R(%)	F(%)
TIT with softmax	91.7	88.6	90.1
Token Interaction Tagger	93.5	90.4	91.9

Table 7: Performance of the token interaction tagging based on the token interaction tagger (TIT) of different structures.

Effectiveness of Token Interaction Tagger. We compare the softmax with CRF as the output layer of the token interaction tagger, and the experimental results are shown in Table 7. We can see that the tagger with CRF can effectively recognize the token interaction and surpass the tagger with softmax by 1.8%. The main reason is that the CRF can utilize the connection of the current tag and the previous tag, where the softmax cannot. Therefore, we conclude that the CRF-based model is more suitable for token interaction tagger.

4 Related Work

Many methods have been proposed for nested NER. Early works on dealing with nested entities rely on hand-craft features or rule-based post-processing (Zhang et al., 2004; Zhou et al., 2004). They use the supervised method that combines the Hidden Markov Model with rule-based post-processing to extract both the inner and outer entities. Moreover, Finkel and Manning (2009) propose a chart-based parsing method for handling nested entities. They construct a discriminative constituency tree to represent each sentence, and each entity is represented as one of the subtrees. However, their method has a cubic time complexity.

Traditionally, the conventional NER is considered as a sequence labeling task. Some studies reveal that sequence labeling-based methods can also perform well on the nested NER. Muis and Lu (2017) introduce a novel notion of mention separators that can effectively detect the nested entity mention. Their method labels gaps between words to yield better performance, which relies on handcrafted features. Ju et al. (2018) propose dynamically stacking flat NER layers, while the number of stacked layers depends on the level of entity nesting. It can recognize entities sequentially from inner to outer. However, their method inevitably suffers from the error propagation since the outer entity detection overly depends on whether the inner entity is correctly recognized or not. Zheng et al. (2019) propose a boundary-aware neural model that leverages entity boundaries to predict entity categorical labels. Their method modifies the BIEO (i.e., Beginning, Internal, End and Other) hypothesis for detecting the boundary of nested entity.

More recently, Lu and Roth (2015) present a novel hypergraph-based method with linear time complexity to tackle the problem of nested entity mention detection. One issue in their approach is the spurious structures of the hypergraph. Wang and Lu (2018) improve the method of Lu and Roth (2015) by modeling arbitrary combinations of mentions with a segmental hypergraph. However, such an architecture leads to a higher time complexity during both training and decoding. Katiyar and Cardie (2018) propose a hypergraph-based representation based on the BILOU tagging scheme. They treat the hypergraph construction procedure as a multi-label assignment process.

5 Conclusions

In this paper, we propose a novel neural model HIT for recognizing nested named entity. It leverages the head-tail pair and token interaction to express the entities with the nested structure. Specifically, the head-tail detector can detect the head-tail pair of named entities. Furthermore, the token interaction tagger captures the internal token connection within the boundary. Experiments on three public datasets show that our model achieves significant improvements over the state-of-the-art models. For future work, we will apply HIT to other languages, and further explore potential cases of overlapping entities in nested NER task.

Acknowledgments

This work was partially supported by Natural Science Foundation of China (No.61772284), Postgraduate Research & Practice Innovation Program of Jiangsu Province (SJKY19_0763). And the Hanghang Tong author is partially supported by NSF (1947135, 1715385, and 1939725).

References

Darina Benikova, Chris Biemann, and Marc Reznicek. 2014. Nosta-d named entity annotation for german: Guidelines and dataset. In *Proceedings of the Ninth International Conference on Language Resources and Evaluation (LREC-2014)*, pages 2524–2531.

Jason PC Chiu and Eric Nichols. 2016. Named entity recognition with bidirectional lstm-cnns. *Transactions of the Association for Computational Linguistics*, 4:357–370.

Timothy Dozat and Christopher D Manning. 2016. Deep biaffine attention for neural dependency parsing. *arXiv preprint arXiv:1611.01734*.

Jenny Rose Finkel and Christopher D Manning. 2009. Nested named entity recognition. In *Proceedings of the 2009 Conference on Empirical Methods in Natural Language Processing*, pages 141–150.

Mourad Gridach. 2017. Character-level neural network for biomedical named entity recognition. *Journal of biomedical informatics*, 70:85–91.

Meizhi Ju, Makoto Miwa, and Sophia Ananiadou. 2018. A neural layered model for nested named entity recognition. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers)*, pages 1446–1459.

Arzoo Katiyar and Claire Cardie. 2018. Nested named entity recognition revisited. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers)*, pages 861–871.

J-D Kim, Tomoko Ohta, Yuka Tateisi, and Jun'ichi Tsujii. 2003. Genia corpus—a semantically annotated corpus for bio-textmining. *Bioinformatics*, 19:180–182.

Jin-Dong Kim, Tomoko Ohta, Yoshimasa Tsuruoka, Yuka Tateisi, and Nigel Collier. 2004. Introduction to the bio-entity recognition task at jnlpba. In *Proceedings of the international joint workshop on natural language processing in biomedicine and its applications*, pages 70–75. Citeseer.

Diederik P Kingma and Jimmy Ba. 2014. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*.

John Lafferty, Andrew McCallum, and Fernando CN Pereira. 2001. Conditional random fields: Probabilistic models for segmenting and labeling sequence data. In *Proceedings of the International Conference on Machine Learning*.

Guillaume Lample, Miguel Ballesteros, Sandeep Subramanian, Kazuya Kawakami, and Chris Dyer. 2016. Neural architectures for named entity recognition. In *Proceedings of the 2016 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 260–270.

Tsung-Yi Lin, Priya Goyal, Ross Girshick, Kaiming He, and Piotr Dollár. 2017. Focal loss for dense object detection. In *Proceedings of the IEEE international conference on computer vision*, pages 2980–2988.

Ying Lin, Shengqi Yang, Veselin Stoyanov, and Heng Ji. 2018. A multi-lingual multi-task architecture for low-resource sequence labeling. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 799–809.

Wei Lu and Dan Roth. 2015. Joint mention extraction and classification with mention hypergraphs. In *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing*, pages 857–867.

Aldrian Obaja Muis and Wei Lu. 2017. Labeling gaps between words: Recognizing overlapping mentions with mention separators. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*, pages 2608–2618.

Jingbo Shang, Liyuan Liu, Xiaotao Gu, Xiang Ren, Teng Ren, and Jiawei Han. 2018. Learning named entity tagger using domain-specific dictionary. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 2054–2064.

Carlos N Silla and Alex A Freitas. 2011. A survey of hierarchical classification across different application domains. *Data Mining and Knowledge Discovery*, 22(1-2):31–72.

Mohammad Golam Sohrab and Makoto Miwa. 2018. Deep exhaustive model for nested named entity recognition. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 2843–2849.

Nitish Srivastava, Geoffrey Hinton, Alex Krizhevsky, Ilya Sutskever, and Ruslan Salakhutdinov. 2014. Dropout: a simple way to prevent neural networks from overfitting. *The journal of machine learning research*, 15(1):1929–1958.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. In *Advances in Neural Information Processing Systems*, pages 6000–6010.

Bailin Wang and Wei Lu. 2018. Neural segmental hypergraphs for overlapping mention recognition. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 204–214.

Jie Zhang, Dan Shen, Guodong Zhou, Jian Su, and Chew-Lim Tan. 2004. Enhancing hmm-based biomedical named entity recognition by studying special phenomena. *Journal of biomedical informatics*, 37(6):411–422.

Changmeng Zheng, Yi Cai, Jingyun Xu, Ho-fung Leung, and Guandong Xu. 2019. A boundary-aware neural model for nested named entity recognition. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing*, pages 357–366.

Guodong Zhou, Jie Zhang, Jian Su, Dan Shen, and Chewlim Tan. 2004. Recognizing names in biomedical texts: a machine learning approach. *Bioinformatics*, 20(7):1178–1190.