IEEE Access
Multidisciplinary | Rapid Review | Open Access Journal

Date of publication xxxx 00, 0000, date of current version xxxx 00, 0000.

Digital Object Identifier 10.1109/ACCESS.2017.DOI

Multilayer ToI Detection Approach for Nested NER

LIN SUN[1,2], (Member, IEEE), FULE JI[†1], KAI ZHANG[1], and CHI WANG[3]
[1]School of Computer and Computing Science, Zhejiang University City College, Hangzhou, China
[2]Intelligent Plant Factory of Zhejiang Province Engineering Lab, Zhejiang University City College, Hangzhou, China
[3]College of Computer Science and Technology, Zhejiang University, Hangzhou, China

Corresponding author: Lin Sun (e-mail: sunl@zucc.edu.cn).

† Fule Ji is a co-first author.
This work was supported in part by the Natural Science Foundation of Zhejiang Province under Grant No. LY17F020008.

ABSTRACT Nested entities commonly exist in news articles and biomedical corpora. The performance of nested NER is still a great challenge in the field of named entity recognition (NER). Unlike the structural models in previous work, this paper presents a comprehensive study of nested NER by means of text-of-interest (ToI) detection. This paper presents a novel ToI-CNN with dual transformer encoders (ToI-CNN+DTE) model for this solution. We design a directional self-attention mechanism to encode contextual representation over the whole-sentence in the forward and backward directions. The features of the entities are extracted from the contextual token representations by a convolutional neural network. Moreover, we use HAT pooling operation to convert the various length ToIs to a fixed length vector and connect with a fully connected network for classification. The layer where the nested entities are located can be evaluated by multi-task learning jointly with layer classification. The experimental results show that our model achieves excellent performance in F1 score, training cost and layer evaluation on the nested NER datasets.

INDEX TERMS nested NER, information extraction, Transformer encoder, convolutional neural network, pooling operation.

I. INTRODUCTION

NER is a fundamental and important natural language processing (NLP) task. It provides input to high level information extraction (IE) such as coreference resolution, relation extraction. In recent years, nested NER has become an active topic in the research of NER [1]–[4]. The nested named entities are common linguistic phenomena in natural language processing (NLP). Nested entity structures frequently exist in broadcast transcripts [5] and biomedical documents [6], such as "Bank of China" and "president of the United States". For example, approximately 17% of the entities in the GENIA corpus are embedded within another entity; 30% of sentences contain nested entities in the ACE corpora. Nested entity can be used to capture finer-grained semantic information. For example, "Bank of China", which is an organization with nested location. Nested NER can help us to recognize that Bank of China is located at China. Although the accuracy of flat NER achieves excellent performance [7], the performance of nested NER is still a great challenge [8]. F1 score of state-of-the-art nested NER methods is approximately 74% on all entities. Actually, after removing flat entities, the performance of F1 score on only nested ones are lower than 60% on ACE datasets and 50% on GENIA dataset, respectively. Figure 1 shows an example of nested NER. Most methods of flat NER is based on conditional random field (CRF) method. However, CRF can only tag one token with a single label and ignores the inner nested ones.

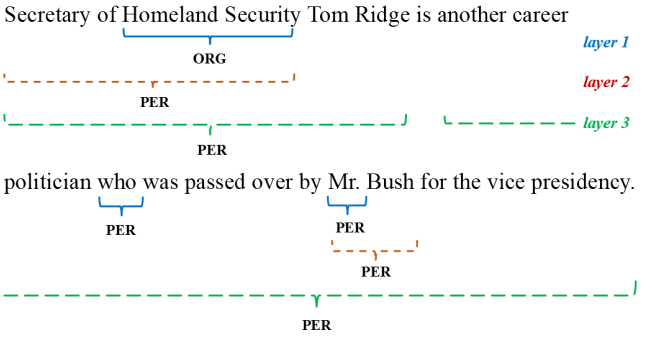

FIGURE 1. An example of nested NER. PER and ORG are entity types. Layer 1, 2, and 3 indicate the layer where the entities are located.

Previous nested NER methods such as parse tree [9], stacking flat layer [2], and hypergraph [3], [13] modeled the

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/ACCESS.2019.2961118, IEEE Access

	Nested representation	Features	Model	Dataset
Finkel and Manning [9]	parse tree	words, POS tags, etc	CRF-CFG [10]	GENIA AnCora
Lu and Roth [11]	hypergraph	words, POS tags, n-grams, etc	mention hypergraph	GENIA ACE2004 ACE2005
Muis and Lu [1]	hypergraph	words, POS tags, n-grams, etc	mention separators + hypergraph	GENIA ACE2004 ACE2005
Ju et al. [2]	stacking flat layer	word embeddings, character embeddings	Bi-LSTM + CRF	GENIA ACE2005
Katiyar and Cardie [3]	hypergraph	word embeddings	Bi-LSTM + hypergraph	GENIA ACE2004 ACE2005
Wang et al. [12]	forest structure	word embeddings, character embeddings, POS tags	stacked LSTM	GENIA ACE2004 ACE2005
Wang and Lu [13]	segmental hypergraph	word embeddings, character embeddings, POS tags	segmental hypergraph	GENIA ACE2004 ACE2005
Lin et al. [14]	anchor + region	word embeddings, character embeddings, POS tags	anchor-region networks	GENIA ACE2005
Merge and Label [15]	split/merge	word embeddings	structure layer + update layer + output layer	ACE2005
ToI-CNN+DTE	text-of-interest	word embeddings, character embeddings, POS tags	dual Transformer encoders + CNN + HAT pooling	GENIA ACE2004 ACE2005

TABLE 1. Comparison of state-of-the-art nested NER methods.

hierarchical structures of the nested entity. In this paper, we propose a ToI-CNN model with dual Transformer encoders to detect nested entity. The contributions of this paper can be summarized as follows:

- Unlike previous methods, we investigate a solution for nested NER by means of text-of-interest detection. ToI detection is a kind of sliding window approach over a sentence. To the best of our knowledge, this work represents the first attempt on nested NER datasets.
- A novel ToI-CNN+DTE model with HAT pooling operation is presented in an end-to-end fashion. The model passes bottom-up ToIs through the contextual and CNN network to encode the context of the sentences and extract the features of the entities, respectively. Moreover, we design HAT pooling operation to obtain a fixed length vector regarding border features for ToI classification.
- We perform a set of extensive experiments on the nested NER datasets. Our model is not complex but effective, and fast in training. The results show the performance of ToI-CNN+DTE in detail and our model achieves excellent performance for nested NER.

II. RELATED WORK

Several methods have been proposed for nested named entity recognition, as shown in Table 1. Finkel and Manning [9] proposed a constituency parser with constituents for each named entity in a sentence. The model employed handcrafted features which were local and pairwise features. The drawback of parse tree is that the complexity is cubic in the number of words and it is time-consuming. Ju et al. [2] and Nguyen et al. [17] designed the stacking multiple layers of LSTM+CRF to detect nested entities. A flat layer consisted of a bidirectional long short-term memory (LSTM) layer and a conditional random field (CRF) layer. One bidirectional LSTM layer represented a word sequence and CRF layer predicted the label sequences in the corresponding nested level. Since one nested layer should have a LSTM+CRF layer, the network would be more complex when the number of the nested layer increased. Due to the limitation of parallelization for LSTM, the training cost of layered LSTM was large.

Another typical nested representation method is hypergraph. Lu and Roth [11] proposed a mention hypergraph model for recognizing overlapping mentions. Then they incorporated mention separators to tag the gap between two words [1]. Recently, Wang and Lu [13] proposed segmental hypergraph representation to model nested entity mentions with pre-trained embeddings. Katiyar and Cardie [3] presented a standard LSTM-based sequence labeling model to learn the nested entity hypergraph structure for an input sentence. The challenge of hypergraph is overlapping mentions in prediction. Although Wang and Lu [13] reduced overlapping mentions, the schema should be carefully designed.

In 2019, Lin et al. [14] proposed anchor-region networks

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/ACCESS.2019.2961118, IEEE Access

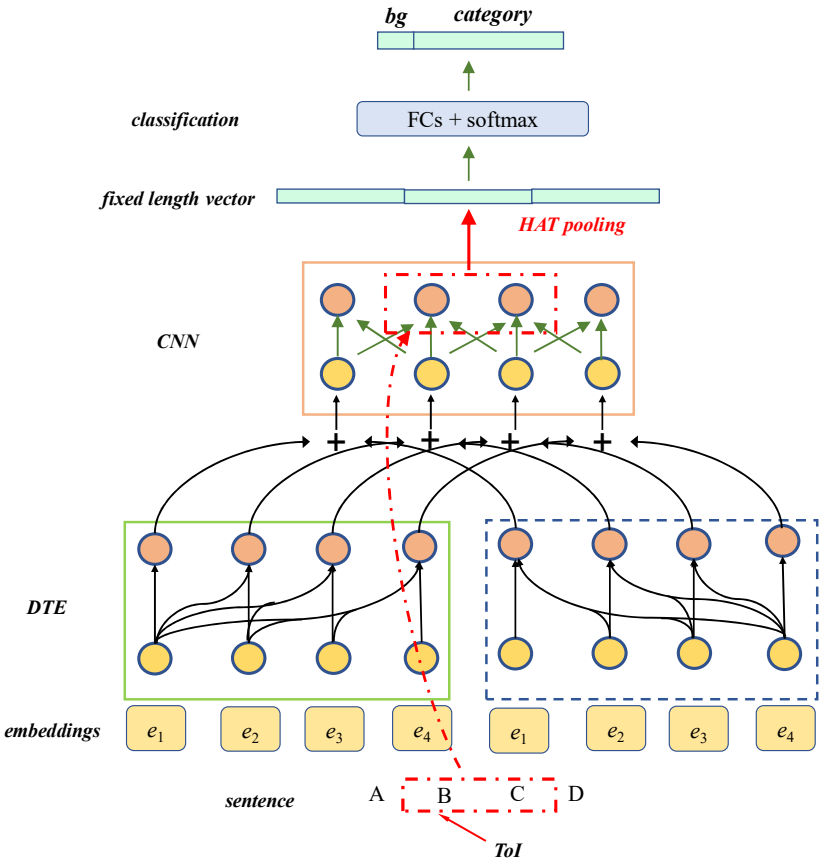

FIGURE 2. Illustration of ToI-CNN+DTE model. To emphasize the relation between the tokens, the DTE only shows the directional self-attention which is the key part of the Transformer encoder. The complete structure about the Transformer encoder can refer to [16]. The symbol "+" indicates that the input of CNN is the sum of two transformer encoders' outputs.

(ARNs) for nested mention detection. ARNs identify head words of mentions and recognize mention regions by exploiting regular phrase structure. Fisher and Vlachos [15] merged tokens into entities forming nested structures and labeled them independently. Compared to classical methods such as LSTM and CRF, the authors tried other ways to solve nested NER, but the performance has hardly been improved.

In the medical domain, Sohrab and Miwa [4] presented a neural exhaustive model and evaluated their model on the biomedical corpus. The model was built on a shared bidirectional LSTM layer and classified potential entity mentions. This model achieved a low recall of 64% due to the excessive negative samples. Marinho et al. [18] proposed a hierarchical nested NER model based on a transition-based parser. In contract information extraction, Sun et al. [19] proposed a single CNN with max pooling to tag lengthy entity mentions in insurance policies. However, the results on ACE and GENIA datasets were not tested in their works.

To address the above issues, we propose a novel ToI-based method to model the contextual dependencies, reduce the training time, eliminate overlapping mentions in prediction, and achieve the excellent performance.

III. TOI-CNN+DTE MODEL

Figure 2 illustrates the model of ToI-CNN with dual Transformer encoders (ToI-CNN+DTE). To highlight the relation between the token representations of input and output, we use arrow lines to connect the circles that represent the tokens. The ToI-CNN+DTE contains four major parts: dual Transformer encoders, CNN, HAT pooling operation, and classification layer. Transformer encoder is based on the original implementation described in [16]. We revise the self-attention part of the Transformer encoder using a directional mask such as left-to-right (green solid line in DTE) and right-to-left (blue dashed line in DTE), respectively. Our model is based on text-of-interest (ToI) detection. The model passes bottom-up ToIs and classifies them as shown in the red dashed rectangle "B C". The CNN layer consists of m filters with kernel size of $k \times d$, where d is the dimension of embeddings and $k = 3$. ToIs from the same sentence share the contextual tokens and extracted feature maps. HAT pooling operation converts the varied length ToIs into a fixed length vector. Then, the fixed length vector can be fed into fully connected (FC) layers and then a softmax layer, which produces the probabilities over $T + 1$ classes, where T is the number of the entity type. The input representation of the token sequence, $\{e_i\}$, is the combination of word embeddings,

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/ACCESS.2019.2961118, IEEE Access

character-level word embeddings, and POS tag embeddings. We use a bidirectional LSTM to construct character-level word embeddings [20].

A. DUAL TRANSFORMER ENCODERS

Vaswani et al. [16] proposed the transformer architecture that is based on the self-attention mechanism. The self-attention mechanism [21], [22] can learn long-distance dependencies and be implemented in efficient parallelization. We use the encoder layer of the Transformer to encode the contextual representations of the words over a sentence. The encoder layer consists of multi-head dot-product self-attention and a fully connected feed-forward layer, coupled with a layer normalization and residual connection. Due to lack of directional information, Vaswani et al. used the absolute position of the tokens as positional encodings at the inputs of the encoder and decoder stacks [16].

In this work, we design a directional self-attention mechanism to emphasize the directional dependency among the input tokens. The directional self-attention is defined as follows:

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

where matrices Q, K, and V consist of queries and keys of dimension d_k, and values of dimension d_v. The details can refer to [16]. Let M denote the directional mask, $M = \begin{bmatrix} 1 & 0 & \dots & 0 \\ 1 & 1 & \dots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 1 & 1 & \dots & 1 \end{bmatrix}$ when the direction is from left to right and $M = \begin{bmatrix} 1 & 1 & \dots & 1 \\ 0 & 1 & \dots & 1 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \dots & 1 \end{bmatrix}$ when the direction is from right to left.

The encoding procedure of dual Transformer encoders is similar to that of bi-LSTM which processes the input from left to right (forward) and right to left (backward), respectively. We place these two opposite directional encoders for the input sentence "A B C D" in Figure 2. The input of the convolutional layer is the sum of two transformer encoders' output.

B. THE CONVOLUTIONAL LAYER

The input of the convolutional layer is $n \times d$, where d is the dimension of embeddings and n is the length of a sentence. The convolution layer has a set of filters $w \in \mathbb{R}^{k \times d}$ and produces m feature maps $f \in \mathbb{R}^{n \times m}$. Rectified linear unit (ReLU) activation is computed after the convolution,

$$f = \sigma(w * x + b), \quad (2)$$

where * denotes the convolution operator, $b \in \mathbb{R}^{m \times 1}$ is a bias term, and σ denotes ReLU activation. We pad $\frac{k-1}{2}$ zeros evenly in the left and right of a sentence to ensure that the size of the output is same as the input.

C. HAT POOLING AND CLASSIFICATION LAYER

The HAT pooling layer uses pooling technique to convert the feature maps of ToIs with any length into a fixed length feature. The pooled out vector is composed of head, average, and tail features (called HAT pooling), as shown in Figure 3. $f = \{f_i\}$ is a convolutional feature. The red dashed rectangle is a ToI feeding forward through the pooling layer. Assume that the position of ToI in f is from i to j, the pooled out vector is $\left[f_i; \frac{1}{j-i+1}\sum_{k=i}^{j} f_k; f_j\right]$. The pooling results with fixed length $3 \times m$ are sequentially arranged into a vector, which is followed by the fully connected layer.

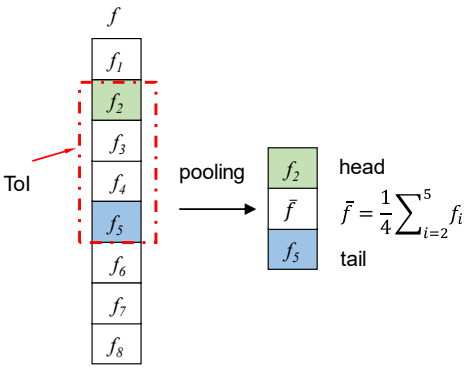

FIGURE 3. Illustration of HAT pooling operation. $\{f_i\}$ are the CNN features of the tokens in a sentence and let $\{f_2, .., f_5\}$ denote a ToI. The HAT pooling result of the ToI is a vector with fixed length $3 \times d$, including head, average $\bar{f}$, and tail.

We perform the classification task on two fully connected (FC) layers and a softmax layer. In FC network,

$$y = \sigma(W^T x + b), \quad (3)$$

where $y \in \mathbb{R}^{v \times 1}$ is the output of FC for given $x \in \mathbb{R}^{u \times 1}$, $W \in \mathbb{R}^{u \times v}$ is a weight matrix, and $b \in \mathbb{R}^{v \times 1}$ is a bias term. The final output of FCs is converted to probability distribution by softmax layer, which is a normalized exponential function. As usual, this layer outputs a probability distribution $p = (p_0, ..., p_T)$. One additional category is negative background, which is defined in the next subsection.

D. MULTILAYER EXTENSION

Several structural models of nested NER as shown in Table 1 can detect the layer of nested entities, in addition to category label. Ju et al. [2] reported the results of layer detection. For layer evaluation in our model, we extend layer classification network with multi-task learning, as shown in Figure 5. In layer classification, three layers need to be identified because the entities above the third layer are almost none [2].

The multi-task loss $\mathcal{L}$ of ToI-CNN+DTE model is the sum of cross-entropy loss on category and layer classification. We do not calculate the loss of background negative in layer classification because the layer of background negative is unknown. The loss function is defined as follows:

$$\mathcal{L} = \mathcal{L}_{category} + \lambda \mathcal{L}_{layer}, \quad (4)$$

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/ACCESS.2019.2961118, IEEE Access

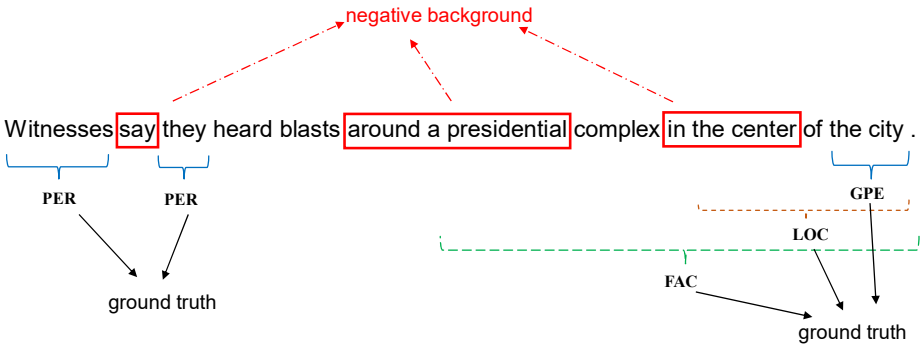

FIGURE 4. Example of labeled ground truths and negative backgrounds. PER, GPE, LOC, and FAC are entity types.

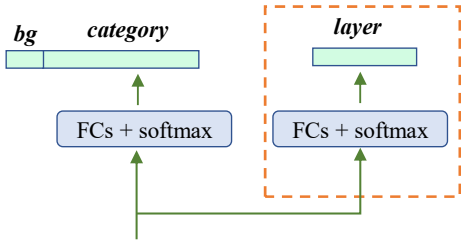

FIGURE 5. Multi-task learning jointly with layer classification, denoted as an orange dashed rectangle.

where λ is a weight parameter.

E. TRAINING SET

In our model, the training set contains two categories: 1) labeled ground truth, 2) negative background. Labeled ground truths consist of all nested level entities. The negative backgrounds are generated from text sliding windows. For example, given a sequence $A\ B\ C$, text sliding windows are $\{A, B, C, AB, BC, ABC\}$. A text sliding window b is considered as negative background, if

$$IoU(b, g^b) \leq IoU_b, \tag{5}$$

where IoU_b is a threshold, g^b has the maximum IoU with b for all ground truths G in one sentence,

$$g^b = \arg\max_{g \in G} IoU(b, g). \tag{6}$$

$IoU(a, b)$ is a function measuring how much overlap occurs between two text strings a and b, and defined as follows:

$$IoU(a, b) = \frac{length(a \cap b)}{length(a \cup b)}. \tag{7}$$

We generate all possible negative backgrounds based on IoU_b with length limitation L_b. Figure 4 illustrates an example of labeled ground truths and negative backgrounds. All labeled entities {Witnesses[PER], they[PER], a presidential complex in the center of the city[FAC], the center of the city[LOC], the city[GPE]} are marked. Given three sliding windows {say, around a presidential, in the center} denoted by red rectangle, the maximum IoU with all ground truths is 0, 1/5, and 1/3 respectively. If we set $IoU_b = 1/2$, they are all considered as negative backgrounds.

IV. EXPERIMENTS

A. DATASETS

We perform nested NER experiments on GENIA [6] and ACE corpora [1]. These datasets and train/dev/test splitting are briefly introduced as follows:

- **GENIA**: This dataset is for biomedical NER. It involves 36 fine-grained entity categories of total 2000 MEDLINE abstracts. Following the same task settings as the previous works, we collapse all DNA, RNA, and protein subcategories into DNA, RNA, and protein respectively, keep cell line and cell type, and remove other types. We split the dataset into train/dev/test sets sequentially with the ratio of 8:1:1 as same as the previous works [1], [9]. We use the dev set to tune parameters. In evaluating the test set, following train/test split of 9:1 in [3], [12], [23], we concatenate train and dev portions as the train set for direct comparison.
- **ACE**: The data of ACE is from newswire and broadcast news. It contains 7 fine-grained entity categories, which are Person (PER), Organization (ORG), Geographical Entities (GPE), Location (LOC), Facility (FAC), Weapon (WEA) and Vehicle (VEH). Testing is performed on the English portion of ACE2004 and ACE2005. We split ACE2004 ad ACE2005 into train/dev/test sets randomly with the ratio of 8:1:1 respectively as same as the previous work [1], [3].

B. SETTINGS

We use GloVe [24] 100-dimensional word vectors for ACE and biomedical word vectors [2] for GENIA. We also update GloVe word embeddings on training via backpropagation [25]. The size of mini-batches is $N_s = 8$ sentences of the same length. For each sentence, we use all labeled ground truths and negative backgrounds for training. In testing, we input all text sliding windows which length is not greater than

[1] https://catalog.ldc.upenn.edu/LDC2005T09 (ACE2004) and https://catalog.ldc.upenn.edu/LDC2006T06 (ACE2005)

[2] wikipedia-pubmed-and-PMC-w2v in the URL of http://bio.nlplab.org/.

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/ACCESS.2019.2961118, IEEE Access

TABLE 2. Hyperparameters of ToI-CNN+DTE.

Parameters	Value
char embedding size	25
char LSTM hidden size	50
transformer encoder stack size	2
number of self-attention heads	4
CNN feature maps	36
CNN kernel height	3
λ	0.2
output dimension of the first FC	64
dropout rate	0.5
learning rate	3-e4
optimizer	Adam
L_b	10
N_s	8
IoU_b for ACE	2/3
IoU_b for GENIA	3/4

TABLE 3. The number of sentences (S), entities (E) and negative backgrounds (B).

		S	E	B
ACE2004	Train	6799	21593	1045877
	Dev	829	2422	134565
	Test	879	2949	145184
ACE2005	Train	7336	24113	1038601
	Dev	958	3172	141453
	Test	1047	2958	135928
GENIA	Train	14824	45973	3645114
	Dev	1885	4963	453177
	Test	1854	5555	476100

L_b for entity detection. Most of the object detection methods use non-maximum suppression (NMS) to reduce the number of false positives [26]. In our ToI detection approach, NMS is not applied because we find that the detection results of ToI-CNN+DTE have no overlappings except nested.

Hyperparameters of ToI-CNN+DTE are shown in Table 2. Statistics of samples on the train/dev/test sets is listed in Table 3. We implement ToI-CNN+DTE network using PyTorch and run Adam [27] optimizer with the learning rate of 3e-4. Autograd in PyTorch computes all the back propagation gradients automatically. After first 10 epochs, we decrease the learning rate by 10% if the training loss increases. The code and trained model of ToI-CNN+DTE on GitHub[3].

C. RESULTS OF TOI-CNN+DTE

Micro-precision (P), micro-recall (R) and micro-F1 score are used for the evaluation metrics in the experiments. Table 4 shows the performance of ToI-CNN+DTE on the development and test sets of ACE2004, ACE2005 and GENIA without extra data and pre-trained model. The performance on the dev and test sets are very close. The difference between the dev and test sets are less than 0.5% in F1 score. Table 5, Table 6 and Table 7 show the categorical performances of ToI-CNN+DTE on test sets of ACE2004, ACE2005 and GENIA respectively. On the ACE datasets, the number of PER entities is the largest and the performance of PER is better than that of other types. On the GENIA dataset, our model performs best in RNA. The number of RNA is the second largest and smaller than that of Protein.

[3]https://github.com/Nested-NER/multilayer-ToI-CNN-DTE

TABLE 4. Performance of ToI-CNN+DTE on the development and test sets.

	dev			test		
	P (%)	R (%)	F1 (%)	P (%)	R (%)	F1 (%)
ACE2004	79.0	74.0	76.4	79.6	74.1	76.8
ACE2005	79.6	72.7	76.0	78.7	73.8	76.2
GENIA	76.9	75.4	76.1	77.4	74.9	76.2

TABLE 5. Categorical performances on the ACE2004 dataset.

Entity type	P (%)	R (%)	F1 (%)
PER	85.5	78.9	82.1
LOC	68.5	58.1	62.9
ORG	70.9	67.2	69.0
GPE	78.9	79.9	79.4
VEH	60.9	82.4	70.0
WEA	47.8	34.4	40.0
FAC	63.2	32.1	42.6
Overall	79.6	74.1	76.8

TABLE 6. Categorical performances on the ACE2005 dataset.

Entity type	P (%)	R (%)	F1 (%)
PER	82.2	81.3	81.7
LOC	65.1	51.9	57.7
ORG	71.0	57.6	64.8
GPE	77.4	79.7	78.6
VEH	63.5	52.9	57.8
WEA	73.9	68.0	70.8
FAC	66.0	51.5	57.9
Overall	78.7	73.8	76.2

TABLE 7. Categorical performances on the GENIA dataset.

Entity Type	P (%)	R (%)	F1 (%)
DNA	75.7	69.5	72.5
RNA	90.9	76.9	83.3
Cell line	78.8	64.5	71.0
Cell type	77.1	69.3	73.0
Protein	77.5	79.6	78.6
Overall	77.4	74.9	76.2

Figure 6 shows the performance of ToI-CNN+DTE with different IOU_b. The great majority of the entities in ACE2004, ACE2005, and GENIA are 1-5 words. Therefore, we set $IOU_b = \{0, 1/3, 1/2, 2/3, 3/4, 4/5\}$ in the form of a fraction number. If $IOU_b > 0$ in Eqn. 5, the negative backgrounds can overlap with the entities. For example, if $IoU_b = 1/2$, "the center", "of the", and "city" could become the negative samples of the entity "the center of the city". This has the advantage of preventing interference from the subsets of the entity strings in sliding window detection, thereby reducing false positives. From Figure 6, we find that when IoU_b is from 1/2 to 2/3, more subsets of the entity strings are added to the negative set, and F1 score gradually increases. F1 score reaches the highest when $IOU_b = 2/3$ on the ACE datasets, and $IOU_b = 3/4$ on the GENIA dataset.

We also test the performance of multi-layer CNN. The multi-layer CNN consists of N CNN layers. Inspired by

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/ACCESS.2019.2961118, IEEE Access

TABLE 8. Comparison with state-of-the-art models on the ACE datasets.

	ACE2004			ACE2005		
	P (%)	R (%)	F1 (%)	P (%)	R (%)	F1 (%)
Muis and Lu [1]	72.7	58	64.5	69.1	58.1	63.1
Ju et al. [2]	-	-	-	74.2	70.3	72.2
Katiyar and Cardie [3]	73.6	71.8	72.7	70.6	70.4	70.5
Wang and Lu [13]	78.0	72.4	75.1	76.8	72.3	74.5
Lin et al. [14]	-	-	-	76.2	73.6	74.9
Merge and Label [15]	-	-	-	75.1	74.1	74.6
ToI-CNN+DTE (HAT pooling)	**79.6**	**74.1**	**76.8**	78.7	**73.8**	**76.2**
ToI-CNN+TE (HAT pooling)	79.2	73.3	76.1	**79.7**	72.0	75.6
ToI-CNN+DTE (average pooling)	78.5	72.9	75.6	75.5	73.5	74.5

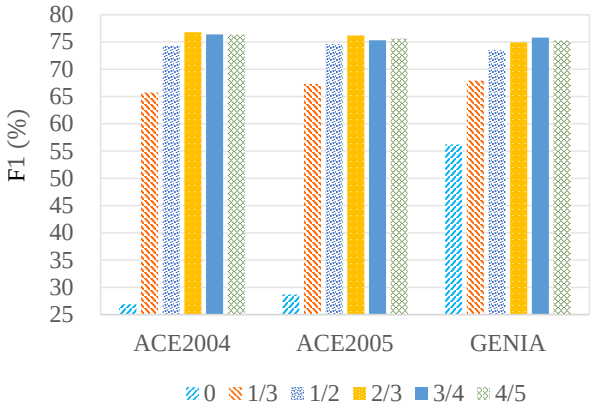

FIGURE 6. Performance of ToI-CNN+DTE with different IOU_b.

ResNets [28], we use a shortcut connection to create a direct path for propagating information. The identical mapping is added to the output of each CNN layer. One CNN layer has d feature maps if the contextual token representation dimension is d so that the input and output of a CNN layer are in the same dimension. We randomly train the model 10 times and the average performance with different depth N on the test sets are shown in Table 9. F1 scores of $N = 1$ are 0.4%, 0.7%, 0.5% higher than those without CNN, i.e. $N = 0$, on the ACE2004, ACE2005, and GENIA datasets, respectively. The performance is slightly changed, approximately $\pm 0.1\%$, when $N > 1$.

TABLE 9. Performance of ToI-CNN+DTE with different depth N.

Depth N	ACE2004	ACE2005	GENIA
0	76.2	75.3	75.1
1	76.6	**76.0**	**75.6**
2	76.7	75.8	75.4
3	**76.8**	75.9	75.5
4	**76.8**	**76.0**	**75.6**

D. COMPARISON WITH STATE-OF-THE-ART MODELS

Table 8 shows the comparison with state-of-the-art models on the ACE2004 and ACE2005 datasets. The performance data of the compared methods are retrieved from the original paper. ToI-CNN+DTE with HAT pooling performs the best in precision, recall and F1 score on ACE2004 and ACE2005. The performance of HAT pooling is better than that of average pooling, increasing 1.2% and 1.7% for F1 score on the ACE2004 and ACE 2005 datasets, respectively. It indicates that border information of the entities is useful for nested NER. We also test the performance of ToI-CNN with one transformer encoder. Compared to ToI-CNN+DTE, F1 score of ToI-CNN+TE decreases 0.7% and 0.6% on the ACE2004 and ACE2005 datasets, respectively.

TABLE 10. Comparison with state-of-the-art models on the GENIA dataset.

	P (%)	R (%)	F1 (%)
Muis and Lu [1]	75.4	66.8	70.8
Ju et al. [2]	78.5	71.3	74.7
Katiyar and Cardie [3]	76.7	71.1	73.8
Wang and Lu [13]	77.0	73.3	75.1
Sohrab and Miwa [4]	**93.2**	64.0	**77.1**
Lin et al. [14]	75.8	73.9	74.8
ToI-CNN+DTE (HAT pooling)	77.4	**74.9**	76.2
ToI-CNN+TE (HAT pooling)	77.6	74.3	75.9
ToI-CNN+DTE (average pooling)	76.8	74.1	75.5

Table 10 shows the comparison with state-of-the-art models on GENIA dataset. The performance of ToI-CNN+DTE is the best of all methods except Sohrab and Miwa [4]. The performance of average pooling also illustrates the effect of HAT pooling, an increase of 0.7% in F1 score on the GENIA dataset. F1 score of ToI-CNN+DTE is also 0.3% higher than that of ToI-CNN+TE.

Our model significantly outperforms the other methods in training cost, benefiting from the self-attention and CNN that can process all data in parallel. In the recurrent neural network model, generating a sequence of state h_t depends on the previous state h_{t-1} and the input of position t. This inherently limits parallelization on training. We train the models on one machine with NVIDIA GeForce GTX 1080Ti (GPU) and AMD Ryzen 7 1800X Processor 3.6GHz (CPU). Table 11 shows the overall training time compared with hypergraph [13] and layered LSTM [2] models. To achieve the optimal performance, training takes approximately 2.3 hours on the ACE datasets and 5.5 hours on the GENIA dataset.

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/ACCESS.2019.2961118, IEEE Access

TABLE 11. Comparison of the overall training time for optimal performance (in hours).

	ACE2004	ACE2005	GENIA
Wang and Lu [13]	10.8	12.6	26.2
Ju et al. [2]	4.3	4.5	11.1
ToI-CNN+DTE	2.2	2.4	5.5

E. RESULTS OF LAYER EVALUATION

Our model can be easily extended to layer evaluation, as shown in Figure 5. Table 12, Table 13 and Table 14 show the multilayer performance on the ACE2004, ACE2005 and GENIA datasets, respectively. We compare our multilayer ToI-CNN+DTE with the layered model of Ju et al. [2]. Ju et al. presented the results of the standard and extended evaluation. The standard evaluation requires that the predicated entities should be on the correct flat NER layer, but the extended evaluation does not require this. We use the standard evaluation metric in Table 12, Table 13 and Table 14. Layer evaluation performance of multilayer ToI-CNN+DTE is better than that of stacking flat NER layers on all datasets.

TABLE 12. Multilayer performance on the ACE2004 dataset.

layer	multilayer ToI-CNN+DTE		
	P (%)	R (%)	F1 (%)
1	78.3	77.4	77.9
2	66.2	55.3	60.3
3	55.4	28.5	37.6

TABLE 13. Multilayer performance on the ACE2005 dataset.

layer	multilayer ToI-CNN+DTE			Ju et al. [2]		
	P (%)	R (%)	F1 (%)	P (%)	R (%)	F1 (%)
1	77.1	75.9	**76.5**	74.5	73.4	73.9
2	65.5	53.7	**59.0**	60.3	50.5	55.0
3	59.0	23.7	**33.8**	51.0	24.5	33.1

TABLE 14. Multilayer performance on the GENIA dataset.

layer	multilayer ToI-CNN+DTE			Ju et al. [2]		
	P (%)	R (%)	F1 (%)	P (%)	R (%)	F1 (%)
1	74.9	73.9	**74.4**	72.9	69.8	71.3
2	48.7	38.2	**42.8**	56.9	27.6	37.2
3	0.0	0.0	0.0	0.0	0.0	0.0

V. CONCLUSION

This paper has presented a novel ToI detection model for nested NER. We design an end-to-end model combining CNN and directional Transformer encoders. The HAT pooling and directional self-attention in the transformer encoder improve the performance of our model, especially on the ACE2004 and ACE2005 datasets. The experimental results show that our ToI-based detection approach obtains the best performance on the ACE datasets and the competitive result on the GENIA dataset without extra data and pre-trained model.

REFERENCES

[1] A. O. Muis and W. Lu, "Labeling gaps between words: Recognizing overlapping mentions with mention separators," in Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing, 2017, pp. 2608–2618.

[2] M. Ju, M. Miwa, and S. Ananiadou, "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), vol. 1, 2018, pp. 1446–1459.

[3] A. Katiyar and C. Cardie, "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), vol. 1, 2018, pp. 861–871.

[4] M. G. Sohrab and M. Miwa, "Deep exhaustive model for nested named entity recognition," in Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing, 2018, pp. 2843–2849.

[5] G. R. Doddington, A. Mitchell, M. A. Przybocki, L. A. Ramshaw, S. Strassel, and R. M. Weischedel, "The automatic content extraction (ace) program-tasks, data, and evaluation." in LREC, vol. 2, 2004, p. 1.

[6] J.-D. Kim, T. Ohta, Y. Tateisi, and J. Tsujii, "Genia corpus—a semantically annotated corpus for bio-textmining," Bioinformatics, vol. 19, no. suppl_1, pp. i180–i182, 2003.

[7] J. P. Chiu and E. Nichols, "Named entity recognition with bidirectional lstm-cnns," Transactions of the Association for Computational Linguistics, vol. 4, pp. 357–370, 2016.

[8] V. Yadav and S. Bethard, "A survey on recent advances in named entity recognition from deep learning models," in Proceedings of the 27th International Conference on Computational Linguistics, 2018, pp. 2145–2158.

[9] J. R. Finkel and C. D. Manning, "Nested named entity recognition," in Proceedings of the 2009 Conference on Empirical Methods in Natural Language Processing: Volume 1-Volume 1. Association for Computational Linguistics, 2009, pp. 141–150.

[10] J. R. Finkel, A. Kleeman, and C. D. Manning, "Efficient, feature-based, conditional random field parsing," Proceedings of ACL-08: HLT, pp. 959–967, 2008.

[11] W. Lu and D. Roth, "Joint mention extraction and classification with mention hypergraphs," in Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing, 2015, pp. 857–867.

[12] B. Wang, W. Lu, Y. Wang, and H. Jin, "A neural transition-based model for nested mention recognition," in Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing, 2018, pp. 1011–1017.

[13] B. Wang and W. Lu, "Neural segmental hypergraphs for overlapping mention recognition," in Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing, 2018, pp. 204–214.

[14] H. Lin, Y. Lu, X. Han, and L. Sun, "Sequence-to-nuggets: Nested entity mention detection via anchor-region networks," arXiv preprint arXiv:1906.03783, appear in ACL 2019, 2019.

[15] J. Fisher and A. Vlachos, "Merge and label: A novel neural network architecture for nested ner," arXiv preprint arXiv:1907.00464, appear in ACL 2019, 2019.

[16] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, Ł. Kaiser, and I. Polosukhin, "Attention is all you need," in Advances in neural information processing systems, 2017, pp. 5998–6008.

[17] T.-S. Nguyen and L.-M. Nguyen, "Nested named entity recognition using multilayer recurrent neural networks," in International Conference of the Pacific Association for Computational Linguistics. Springer, 2017, pp. 233–246.

[18] Z. Marinho, A. Mendes, S. Miranda, and D. Nogueira, "Hierarchical nested named entity recognition," in Proceedings of the 2nd Clinical Natural Language Processing Workshop, 2019, pp. 28–34.

[19] L. Sun, K. Zhang, F. Ji, and Z. Yang, "Toi-cnn: A solution of information extraction on chinese insurance policy," NAACL-HLT 2019, pp. 174–181, 2019.

[20] G. Lample, M. Ballesteros, S. Subramanian, K. Kawakami, and C. Dyer, "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, 2016, pp. 260–270.

[21] A. P. Parikh, O. Täckström, D. Das, and J. Uszkoreit, "A decomposable attention model for natural language inference," arXiv preprint arXiv:1606.01933, 2016.

[22] Z. Lin, M. Feng, C. N. d. Santos, M. Yu, B. Xiang, B. Zhou, and Y. Bengio, "A structured self-attentive sentence embedding," arXiv preprint arXiv:1703.03130, 2017.
[23] J. Straková, M. Straka, and J. Hajic, "Neural architectures for nested ner through linearization," in Proceedings of the 57th Conference of the Association for Computational Linguistics, 2019, pp. 5326–5331.
[24] J. Pennington, R. Socher, and C. Manning, "Glove: Global vectors for word representation," in Proceedings of the 2014 conference on empirical methods in natural language processing (EMNLP), 2014, pp. 1532–1543.
[25] R. Collobert, J. Weston, L. Bottou, M. Karlen, K. Kavukcuoglu, and P. Kuksa, "Natural language processing (almost) from scratch," Journal of Machine Learning Research, vol. 12, no. Aug, pp. 2493–2537, 2011.
[26] R. Girshick, "Fast r-cnn," in Proceedings of the IEEE international conference on computer vision, 2015, pp. 1440–1448.
[27] D. P. Kingma and J. Ba, "Adam: A method for stochastic optimization," arXiv preprint arXiv:1412.6980, 2014.
[28] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," in Proceedings of the IEEE conference on computer vision and pattern recognition, 2016, pp. 770–778.