HiTRANS: A Hierarchical Transformer Network for Nested Named Entity Recognition

Zhiwei Yang[1,2,3,4], **Jing Ma**[2], **Hechang Chen**[3,4,*], **Yunke Zhang**[3], **Yi Chang**[3,4,*]
[1] College of Computer Science and Technology, Jilin University, Changchun, China
[2] Department of Computer Science, Hong Kong Baptist University, Hong Kong, China
[3] School of Artificial Intelligence, Jilin University, Changchun, China
[4] Key Laboratory of Symbolic Computation and Knowledge Engineering of Ministry of Education
yangzw18@mails.jlu.edu.cn, majing@comp.hkbu.edu.cn,
chenhc@jlu.edu.cn, zhangyk19@mails.jlu.edu.cn, yichang@jlu.edu.cn

Abstract

Nested Named Entity Recognition (NNER) has been extensively studied, aiming to identify all nested entities from potential spans (i.e., one or more continuous tokens). However, recent studies for NNER either focus on tedious tagging schemas or utilize complex structures, which fail to learn effective span representations from the input sentence with highly nested entities. Intuitively, explicit span representations will contribute to NNER due to the rich context information they contain. In this study, we propose a **Hi**erarchical **Trans**former (HiTRANS) network for the NNER task, which decomposes the input sentence into multi-grained spans and enhances the representation learning in a hierarchical manner. Specifically, we first utilize a two-phase module to generate span representations by aggregating context information based on a bottom-up and top-down transformer network. Then a label prediction layer is designed to recognize nested entities hierarchically, which naturally explores semantic dependencies among different spans. Experiments on GENIA, ACE-2004, ACE-2005 and NNE datasets demonstrate that our proposed method achieves much better performance than the state-of-the-art approaches.

1 Introduction

Named entity recognition (NER) is an essential task in the research of natural language processing, which aims to detect and classify text spans into corresponding semantic categories, e.g., Person (PER), Organization (ORG), and Location (LOC), from a chunk of text. Most existing studies focus on flat NER, i.e., without nested entities, by sequence labeling methods (Yang et al., 2020; Yoon et al., 2019). However, named entities are generally nested with each other in the real world (Finkel and Manning, 2009). For example, in Figure 1, the

Figure 1: An example of nested named entities. The short entities with red labels are nested in long entities with blue labels, respectively, in a hierarchical manner.

entity "St. Louis" with Label "CITY" is nested in "St. Louis Cardinals" with Label "SPORTSTEAM". This poses a major technical challenge to the previous methods and thus a more robust model for nested NER (NNER) is urgently desirable.

Previous literature for NNER can be roughly categorized into three types: 1) hypergraph-based models focus on designing a complex hypergraph structure to obtain an expressive tagging schema (Straková et al., 2019; Katiyar and Cardie, 2018; Lu and Roth, 2015), which are time-consuming when encountering ambiguous schemas; 2) span-based models tend to detect candidate spans from an input sentence first, and then train a classifier to predict entity categories (Luo and Zhao, 2020; Zheng et al., 2019). However, it is hard to get a complete meaning of the sentence because each text span contains only a part of the semantics, and errors may propagate to the prediction stage if the span boundary is divided incorrectly at the first stage; and 3) layered-based models are proposed to utilize layered structures to deal with NNER based on the divide and conquer strategy (Jue et al., 2020; Xia et al., 2019; Ju et al., 2018). However, it merely breaks down the complex problem into several smaller subtasks and pays little attention to the hierarchical representation learning for multi-grained named entities.

To this end, we propose a novel hierarchical transformer network (HiTRANS) to recognize more named entities (either nested or not) for a given sentence, where we capture the dependencies of adjacent candidate spans and utilizes an attention mech-

* Corresponding authors.

Findings of the Association for Computational Linguistics: EMNLP 2021, pages 124–132
November 7–11, 2021.

anism to enhance the representation of text spans. More specifically, the input of our proposed method combines character-level, word-level, and sentence-level representations, which are obtained by three embedding networks, respectively. We then propose a two-phase span generation model (SGM) on top of the multi-level representations, which hierarchically aggregates adjacent spans based on transformer mechanism at each layer. The SGM includes a bottom-up and a top-down structure to enhance the representation learning of each candidate span which is finally fed into a label prediction layer to assign an entity class for the span. As a result, nested entities are comprehensively contained in the candidate spans and the representation learning is further enhanced based on both multi-level embedding and the hierarchical transformer mechanisms. Experimental results on FOUR datasets demonstrate that HiTRANS establishes new state-of-the-art performances, which verifies the effectiveness of our proposed framework. The main contributions of this work are as follows:

- We propose a novel hierarchical transformer framework (HiTRANS) for NNER, which is superior in modeling the nested relations among multi-grained named entities and learning more effective representations.
- Entity representation learning is formulated as a two-phase span generation, which aggregates context information of adjacent spans in a bottom-up and top-down manner respectively. The span representation is enhanced by multi-level features and context information.
- The overall superiority of our HiTRANS is validated across four benchmarks comparing with state-of-the-art methods. Visualization and case study conducted on top of the outputs from each layer further shows an in-depth understanding of our method.

2 Related Work

We briefly review some prior works closely related to ours from three perspectives: hypergraph-based, span-based, and layered-based approaches.

Hypergraph-based approaches obtain expressive tagging schemas for NNER (Lu and Roth, 2015; Wang and Lu, 2018). However, the hypergraph requires specific modules designed to prevent the spurious structure of hypergraphs. Muis and Lu (2017) introduced mention separators to facilitate multi-graph representation. Katiyar and Cardie (2018) further improved the result using features extracted from a recurrent neural network. Recently, Straková et al. (2019) proposed two competitive neural networks using a linearized scheme. However, more expressive and unambiguous schemas will inevitably cause higher time complexity.

Span-based methods achieve promising results for NNER (Tan et al., 2020; Zheng et al., 2019; Sohrab and Miwa, 2018), which explicitly enumerate all possible spans from input sentences, which will be fed into a classifier for category prediction based on multitask learning. Lin et al. (2019) proposed a sequence-to-nuggets architecture to recognize nested entities with semantic central words. Li et al. (2020) extracted answer spans from a passage through a given question. Luo and Zhao (2020) proposed a novel bipartite flat-graph network to learn the dependencies of inner spans. But most of these methods generally break input sequences into fragments, leading to inferior semantics.

Layered-based models are recently proposed, e.g., Finkel and Manning (2009) constructed a syntactic constituency tree to transform each sentence into a tree,Wang et al. (2018) proposed a transition-based model by mapping a sentence with nested mentions to a designated forest, Fisher and Vlachos (2019) and Ju et al. (2018) dynamically stacked multiple flat NER layers from inside to outside, Shibuya and Hovy (2020) introduced a decoding method that iteratively recognizes entities in an outside-to-inside way, Jue et al. (2020) and Xia et al. (2019) utilized a layered model to recursively identify named entity candidates based on a hierarchical structure, which is suitable for NNER. However, few of them emphasize on learning more effective span representations, failing to recognize nested named entities in more complex sentences.

The core idea of our work is inspired to enhance representation learning for more complex sentences. We propose to leverage the representation power of transformer based on a hierarchical structure for improving NNER. Particularly, pre-trained word embeddings such as GloVe (Pennington et al., 2014), and pre-trained sentence-level embedding such as BERT (Devlin et al., 2019) and ALBERT (Lan et al., 2020) have proven to be effective to NER. In this paper, we will apply both kinds of embeddings besides character embeddings to further improve the performance.

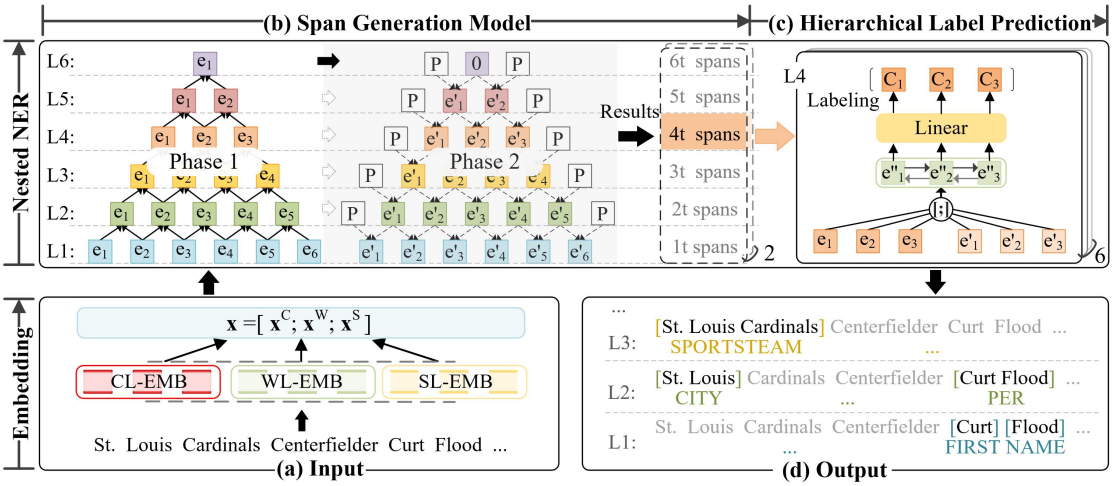

Figure 2: An overview of our HiTRANS model. (a) The character-level embedding (CL-EMB), word-level embedding (WL-EMB), and sentence-level embedding (SL-EMB) are concatenated for better representations. (b) There are two phases in span generation model (SGM) network, which iteratively generate span representations for each layer by merging adjacent spans in a bottom-up and a top-down manner, respectively. "P" denotes the padding when employing CNN and the outputs "*t spans" denote the representations of candidate spans at each layer, where L is set to 6 in the above example. (c) During hierarchical label prediction (HLP), the same labeling network, e.g., $L4$, is employed in each layer. (d) Output entities. Different layers are displayed in different colors.

3 Our Proposed Method

Prior hypergraph-based and span-based methods for NNER suffer from ambiguous schemas or errors propagation in complex sentences, thus layered-based models are proposed to decompose the problem into several smaller subtasks. However, for NNER, learning effective representations and modeling inter-entity dependencies is still a substantial challenge. In this study, we hypothesis that nested entities in the same context are complementary and the text representation at multi-level could improve NNER.

Given an input sentence $\mathcal{S}$ is composed of a sequence of words, i.e., $\mathcal{S} = \{w_1, w_2, \ldots, w_{|\mathcal{S}|}\}$, where $|\mathcal{S}|$ denotes the number of words. For the NNER task, each word w_i is associated with multiple BIO2-format [1] labels $\mathbf{Y}_i = \{\mathbf{Y}_i^1, \mathbf{Y}_i^2, \ldots, \mathbf{Y}_i^L\}$, where L denotes the maximum nesting depth. Note that if $L = 1$, a word w_i is associated with one categorical label, which is regarded as flat NER. Therefore, we formulate NNER as a multi-layer prediction problem. Specifically, the topmost layer is processed as flat NER, and other layers merely using $B-\{class\}$ and O labels to recognize complete entities from text spans. For each layer, it modeled as sequence labeling, that is, $f^l : \mathbf{e}_1\mathbf{e}_2 \cdots \mathbf{e}_T \rightarrow y_1 y_2 \cdots y_T$, where $\mathbf{e_i}$ indicates the representation of a text span (i.e., one or more continuous words) iteratively generated from the previous layer, T indicates the number of spans in the l-th layer ($1 \leq l \leq L$).

In the following subsections, we will introduce our proposed **HiTRANS**, which consists of three parts: Multi-level Representation, Span Generation Model, and Hierarchical Label Prediction. Figure 2 gives an overview of our framework.

3.1 Multi-level Representation

To better capture the semantic information of a sentence, we learn token representations from multiple levels, e.g., character level, word level, and sentence level. As Figure 2 (a) shows, given a sentence composed of a sequence of words $\mathcal{S} = \{w_1, w_2, \ldots, w_{|\mathcal{S}|}\}$ and c_{ij} denotes the j-th character within the i-th word w_i. For the i-th word, the multi-level representation is represented as follows:

$$\mathbf{x}_i = [\mathbf{x}_i^c; \mathbf{x}_i^w; \mathbf{x}_i^s] \quad (1)$$

where $\mathbf{x}_i^c$ denotes the character-level representation within w_i. As each word can be regarded as a character sequence, randomly initialized character embeddings are encoded by a bidirectional LSTM layer (Zheng et al., 2019) to capture sequential features in the context, then we use the last hidden state as $\mathbf{x}_i^c$. $\mathbf{x}_i^w$ denotes the word-level representation obtained from GloVe (Pennington et al., 2014)

[1] *B-, I-, O* indicate the beginning, intermediate, and outer position of an entity, respectively, and *class* indicates a categorical label takes from a pre-defined tag set, e.g., *Person*, *Location*, or *Organization*.

for the i-th word w_i; and $\mathbf{x}_i^s$ denotes the sentence-level representation obtained from pretrained language model, e.g., BERT and ALBERT. And $[;]$ denotes concatenation. Furthermore, a dense layer is applied to reduce the embedding dimension, i.e., $\mathbf{x}_i \rightarrow \mathbf{e}_i$. Thus, we can obtain the span representation in the l-th layer as $\mathbf{H}^l = \{\mathbf{e}_1, \mathbf{e}_2, ..., \mathbf{e}_T\}$, where T is the span number.

In order to learn more effective span representations in Figure 2 (b), we further adapt the multi-head attention mechanism from Transformer (Vaswani et al., 2017) in each layer of HiTRANS, as illustrated in Figure 3. Specifically, in the l-th layer, HiTRANS first transforms the multi-level representation $\mathbf{H}^l$ into multiple subspaces with different linear projections:

$$\mathbf{Q}_h, \mathbf{K}_h, \mathbf{V}_h = \mathbf{H}^l\mathbf{W}_h^Q, \mathbf{H}^l\mathbf{W}_h^K, \mathbf{H}^l\mathbf{W}_h^V \quad (2)$$

where $\{\mathbf{Q}_h, \mathbf{K}_h, \mathbf{V}_h\}$ are respectively the query, key, and value representations with trainable parameters $\{\mathbf{W}_h^Q, \mathbf{W}_h^K, \mathbf{W}_h^V\}$ corresponding to the h-th head. Then, the attention functions are applied to refine the span representations.

$$\mathbf{H}_h^l = softmax(\frac{\mathbf{Q_h}\mathbf{K_h}^T}{\sqrt{d_h}})\mathbf{V}_h \quad (3)$$

where $\mathbf{H}_h^l$ is the h-th head with d_h as the dimension. Furthermore, we concatenate the output representations of all these heads with the residual connection to capture global semantic information in parallel, which is as follows:

$$\mathbf{H}^l = [\mathbf{H}_1^l; \mathbf{H}_2^l; \ldots; \mathbf{H}_n^l]\mathbf{W}^O + \mathbf{H}^l \quad (4)$$

where $\mathbf{H}^l \in \mathbb{R}^{T \times nd_h}$ is the final span representation in the l-th layer, n is the number of parallel heads, and $\mathbf{W}^O$ is a trainable parameter. For example, $\mathbf{H}^1$ indicates the refined span representations for the first layer at Phase 1 of Figure 2 (b).

3.2 Span Generation Model

To extract nested entities from nested-structure sentences, we design a hierarchical span generation model (SGM) consisting of two phases to generate candidate spans for the NNER, as shown in Figure 2 (b). Specifically, the two phases are composed of L layers that respectively generate candidate spans in a Bottom-Up and Top-Down manner (i.e., BU-SGM and TD-SGM) in sequence. In each layer of SGM, a convolution neural network (CNN) is firstly utilized to aggregate two adjacent spans for the next layer which generates all possible flat entities as candidates for further prediction. Then a multi-head attention layer is utilized to enhance the representation learning of each candidate. The details of each component will be described below.

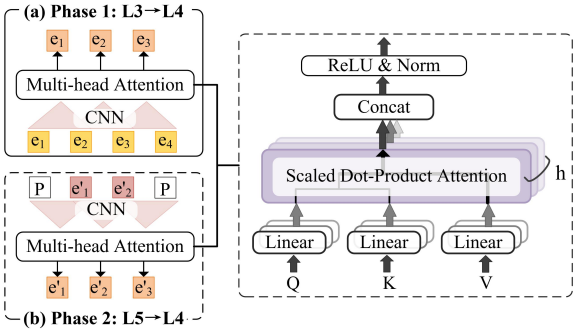

Figure 3: Detailed structure of candidate span generation for Layer 4. (a) the refined span representations from Layer 3 at Phase 1; (b) the refined the representations from Layer 5 at Phase 2.

BU-SGM. The core idea of BU-SGM network is to generate feature vectors for candidate spans by recursively stacking convolutional neural networks from the bottom layer to the top layer as shown at Phase 1 of SGM in Figure 2 (b). Specifically, the generated span representations in the first layer correspond to 1-token entities. As for higher layers, a CNN with a kernel size of 2 is iteratively applied to generate continuous l-token span representations from the $(l-1)$-th layer in a bottom-up manner, which avoids breaking the consecutive context. The span representations in the l-th layer can be obtained in a bottom-up manner:

$$\hat{\mathbf{H}}^l = \begin{cases} f(\mathbf{H}^l) & , l = 1 \\ f(Conv(\hat{\mathbf{H}}^{l-1})) & , 1 < l \leq L \end{cases} \quad (5)$$

where $f(\cdot)$ indicates the shorthand of Equation (4), $\mathbf{H}^l$ denotes the refined multi-level representation obtained from the first layer, and $\hat{\mathbf{H}}^l$ denotes span representations in Layer l generated iteratively from Layer $l-1$. It is noted that stacking CNN will lead to the length reduced by 1 in each layer. Besides, a ReLU and Norm layer is applied to obtain the final span representations.

TD-SGM. In the opposite direction, as long entities at higher-layer are closely related to short entities at lower-layer in the same context, high-level features can contribute to identifying entities in lower layers by providing additional background information, which is complementary with low-level features. Therefore, TD-SGM network aims

to propagate the higher-layer information to lower layers in a top-down manner, which is initialized by $\mathbf{0}$ and guided with the output from the corresponding layer of Phase 1. Specifically, the span representations generated at Phase 2 is iteratively obtained by stacking CNNs (with a kernel size of 2) with proper zero-paddings in a top-down manner. For example, as Figure 2 (b) shows, the span representations in Layer 4 at Phase 2, i.e.$\{\mathbf{e}_1^{'}, \mathbf{e}_2^{'}, \mathbf{e}_3^{'}\}$, are generated from the span representations in Layer 5, i.e., $\{\mathbf{0}, \mathbf{e}_1^{'}, \mathbf{e}_2^{'}, \mathbf{0}\}$, which are obtained by concatenating the span representations in Layer 5 at Phase 1 and Phase 2, and then padding with zeros. Similarly, the span representations in the l-th layer can be generated in a top-down manner as follows:

$$\check{\mathbf{H}}^l = \begin{cases} f([\mathbf{0}; \hat{\mathbf{H}}^l]) & , l = L \\ f(Conv^{'}([\check{\mathbf{H}}^{l+1}; \hat{\mathbf{H}}^{l+1}])) & , 1 \leq l < L \end{cases} \quad (6)$$

where $\check{\mathbf{H}}^l$ denotes span representations in Layer l generated from Layer $l+1$, $\mathbf{0}$ denotes zero tensor for initializing the top-layer representation.

3.3 Hierarchical Label Prediction

To recognize named entities from candidate spans, a hierarchical label prediction (HLP) network is introduced, as shown in Layer 4, Figure 2 (c). First, the outputs of Phase 1 and Phase 2 are concatenated as the final candidate span representations to combine bidirectional features into a global informative representation. Formally, the final span representations in the l-th layer are as follows:

$$\mathbf{H}^l = [\hat{\mathbf{H}}^l; \check{\mathbf{H}}^l] \quad (7)$$

As BiLSTM networks can make full use of the context information at a higher level, we employ a BiLSTM and a linear layer to predict labels for candidate spans in a hierarchical manner. As we have obtained complete candidate spans, e.g., {1-token spans, 2-token spans, ..., L-token spans}, based on the attention weights in the SGM module, we can easily classify them into a proper category. The predicted labels for the span representations in the l-th layer is obtained as follows:

$$\mathbf{H}^l = BiLSTM(\mathbf{H}^l\mathbf{U}_1 + \mathbf{b}_1) \quad (8)$$

$$\mathbf{Y}^l = argmax(\mathbf{H}^l\mathbf{U}_2 + \mathbf{b}_2) \quad (9)$$

where $\mathbf{U}_1, \mathbf{U}_2, \mathbf{b}_1$, and $\mathbf{b}_2$ are trainable parameters, $\mathbf{Y}^l$ is the predicted labels of the l-th layer. The total output of the L layers is $\mathbf{Y} = \{\mathbf{Y}^1, \mathbf{Y}^2, ..., \mathbf{Y}^L\}$.

3.4 Model Training

We prepare the gold labels in a hierarchical manner, therefore, each layer of the proposed model could be simplified as a multi-class classification task in any layer of bottom $L-1$ layers and a flat NER task in the topmost layer. During training, our model predicts the distribution of entity semantic labels for each layer. Finally, we compute the cross-entropy loss as follows:

$$\mathcal{L} = -\sum(\hat{\mathbf{Y}}^l)\log(\mathbf{Y}^l) \quad (10)$$

where $\hat{\mathbf{Y}}^l$ and $\mathbf{Y}^l$ denote the true distribution and predicted distribution of entity semantic labels, respectively. $\mathcal{L}$ is the summation of the loss from all layers. Our complete training procedure for HiTRANS is shown in Algorithm 1.

Algorithm 1 Pseudocode of HiTRANS.

Input: A sequence of words $\mathcal{S} = \{w_1, w_2, \ldots, w_{|\mathcal{S}|}\}$; The number of layers $L(l \in L)$
Output: Entity labels of L layers $\mathbf{Y} = \{\mathbf{Y}^1, \mathbf{Y}^2, ..., \mathbf{Y}^L\}$.
for numbers of training iterations **do**
  Multi-level Embedding $\mathbf{x}_i = [\mathbf{x}_i^c; \mathbf{x}_i^w; \mathbf{x}_i^s]$
  Attention-refined representation for the l−th layer using Equation (2) and (3)
  initializing $\hat{\mathbf{H}}^l \leftarrow$ Equation (4)
  [*Span Generation*]
  for l:=1 **to** L **step** 1 **do**
    BU-SGM: $\hat{\mathbf{H}}^l \leftarrow$ Equation (5)
  for l:=L **to** 1 **step** 1 **do**
    TD-SGM: $\check{\mathbf{H}}^l \leftarrow$ Equation (6)
  [*Hierarchical Label Prediction*]
  for l:=1 **to** L **step** 1 **do**
    $\mathbf{H}^l = [\hat{\mathbf{H}}^l; \check{\mathbf{H}}^l]$
    $\mathbf{H}^l \leftarrow$ BiLSTM by Equation (8)
    $\mathbf{Y}^l \leftarrow$ Equation (9)
end for
return Entity labels $\mathbf{Y}$

4 Experiments

4.1 Datasets and Baseline Methods

To verify the effectiveness of HiTRANS, we conduct our experiments on four NNER datasets: GENIA (Kim et al., 2003), ACE-2004 (Doddington et al., 2004), ACE-2005 (Walker et al., 2006), and NNE (Ringland et al., 2019). We adopt the preprocess of (Finkel and Manning, 2009) and (Lu and Roth, 2015). For GENIA, we use GENIA v3.0.2 corpus to construct the dataset and split it into 81%:9%:10% for training, development, and testing, respectively. For ACE-2004 and ACE-2005, we split the Train/Develop/Test set following the preprocess as previous studies (Lu and Roth, 2015; Zheng et al., 2019; Jue et al., 2020). For NNE, we

Item	GENIA			ACE-2005			NNE			ACE-2004		
	Train	Develop	Test	Train	Develop	Test	Train	Develop	Test	Train	Develop	Test
Total sentences	15022	1669	1855	6198	742	809	7285	968	1058	43457	1989	3762
Nested sentences	3222	328	448	2718	294	388	2797	352	339	28606	1292	2489
Total entities	47006	4461	5596	22195	2514	3034	24700	3218	3029	248136	10463	21196
Nested entities	8382	818	1212	10157	1092	1417	9946	1191	1179	206618	8487	17670
Max length	20	20	15	57	35	42	49	31	27	16	15	15
Percentage	18%	18%	22%	46%	43%	47%	40%	37%	39%	83%	81%	83%

Table 1: The statistics of datasets. A nested sentence denotes the sentence containing any nested entity.

Model	GENIA			ACE-2004			ACE-2005			NNE		
	P(%)	R(%)	F1(%)	P(%)	R(%)	F1(%)	P(%)	R(%)	F1(%)	P(%)	R(%)	F1(%)
Revisited (Katiyar and Cardie, 2018)	79.80	68.20	73.60	73.60	71.80	72.70	70.60	70.40	70.50	-	-	-
Linearization (Straková et al., 2019)	-	-	76.40	-	-	77.10	-	-	75.40	-	-	-
Exhaustive (Sohrab and Miwa, 2018)	93.20	64.00	77.10	70.40	62.50	66.20	75.20	58.00	65.50	-	-	-
Boundary-aware (Zheng et al., 2019)	74.00	76.10	75.00	74.40	74.10	74.20	76.40	71.20	73.70	89.10	89.30	89.20
Sequence2nuggets (Lin et al., 2019)	75.80	73.90	74.80	-	-	-	76.20	73.60	74.90	-	-	-
Boundary-enhanced (Tan et al., 2020)	78.90	72.70	75.70	78.10	72.80	75.30	77.10	74.20	75.60	-	-	-
BiFlaG (Luo and Zhao, 2020)	77.40	74.60	76.00	-	-	-	75.00	75.20	75.10	-	-	-
Layered (Ju et al., 2018)	78.50	71.30	74.70	-	-	-	74.20	70.30	72.20	-	-	-
Second-best (Shibuya and Hovy, 2020)	76.30	74.70	75.50	-	-	-	83.00	82.40	82.70	-	-	-
Multi-grained (Xia et al., 2019)	-	-	-	81.70	77.40	79.50	79.00	77.30	78.20	-	-	-
Merge (Fisher and Vlachos, 2019)	-	-	76.44	-	-	77.08	-	-	75.36	-	-	-
Pyramid (Jue et al., 2020)	78.60	77.02	77.78	81.14	79.42	80.27	80.01	78.85	79.42	93.44	93.95	93.70
Merge (Fisher and Vlachos, 2019)	-	-	78.20	-	-	84.33	-	-	83.42	-	-	-
MRC (Li et al., 2020)	78.56	73.94	76.18	87.39	86.09	86.73	86.90	86.50	86.70	-	-	-
Boundary-enhanced (Tan et al., 2020)	79.20	77.40	78.30	85.80	84.80	85.30	83.80	83.90	83.90	-	-	-
Pyramid (Jue et al., 2020)	-	-	-	87.71	87.78	87.74	85.30	87.40	86.34	94.30	95.07	94.68
HiTRANS	78.57	79.59	**79.08**	88.10	87.57	**87.88**	86.48	87.62	**87.04**	94.62	94.85	**94.74**

Table 2: Experiment results on the test set of four benchmarks compared to the state-of-the-art methods. Methods listed in the lower part of the table are based on the pretrained language model.

use the original dataset split and pre-processing. There are 5/7/7/114 different entity types in GENIA, ACE-2004, ACE-2005, and NNE datasets, respectively. For evaluation, we employ micro-averaged precision (P), recall (R), and F1. Table 1 lists the concerned data statistics of each dataset.

We comprehensively compare our proposed model with the state-of-the-art baselines, which could be categorized into three groups as follows:

- **Hypergraph-based methods**: These obtain expressive tagging schemas for NER, including Revisited Model (Katiyar and Cardie, 2018), and Linearization model (Straková et al., 2019).
- **Span-based methods**: They achieve a decent performance by enumerating possible regions of an input sequence for classification, including Exhaustive Model (Sohrab and Miwa, 2018), Boundary-aware (Zheng et al., 2019), Sequence2nuggets (Lin et al., 2019), MRC (Li et al., 2020), Boundary-enhanced (Tan et al., 2020), and BiFlaG (Luo and Zhao, 2020).
- **Layered-based methods**: These methods apply hierarchical structures to iteratively extract named entities in order, including Layered Model (Ju et al., 2018), Merge (Fisher and Vlachos, 2019), Second-best Model (Shibuya and Hovy, 2020), Multi-grained Model (Xia et al., 2019), and Pyramid (Jue et al., 2020).

4.2 Experimental Settings

We obtain the character-level representation encoded by BiLSTM, and word-level representation from the 100-dimensional pre-trained word embedding GloVe (Pennington et al., 2014), which is trained in 6B tokens. For sentence-level embeddings, we use the BERT and ALBERT embeddings to further improve the NNER. For ACE-2004, ACE-2005, and NNE datasets, the dimensions of character-level embedding, word-level embedding, sentence-level embedding are set by default to 30, 100, and 5120 (1024+4096), respectively. As for the GENIA dataset, we obtain word embedding from pretrained embedding Pubmed trained on biomedical corpus (Chiu et al., 2016), setting the dimension of word-level embeddings to 200. The output dimension of the multi-level representation and the hidden size of bidirection LSTM are set to 200. The number of parallel heads k is set to 8.

The number of layers L is set to 16, which exceeds the length of most entities and the batch size is empirically set to 32. We use SGD optimizer for training our model with learning rate set to 0.02, and the dropout rate is set to 0.4 to avoid overfitting. All of our experiments are performed on the same machine. We repeat these experiments 5 times, and report the average performance on the test set.

4.3 Results and Analysis

Table 2 shows the overall results compared with the baseline methods by groups. Overall, hypergraph-based methods achieve decent results depending on the expressive tagging schema; however, ambiguity and high time complexity are hardly inevitable. Span-based methods improve the performance of NNER; however, they may break the continuous-structure of the context. To alleviate the problem, layered-based models further improve the final performance with hierarchical layers, however, span representations are oversimplified. In addition, the methods incorporated with the pretrained language model, e.g., BERT and ALBERT, generally outperform previous methods, which take the advantage of capturing sentence-level features from context. As shown in Table 1, we can observe that there are 22%, 47%, 39%, and 83% in the test set of GENIA, ACE-2004, ACE-2005, and NNE, respectively, which contain nested entities to different degrees. Table 2 shows that our proposed HiTRANS achieves the state-of-the-art results on GENIA [2], ACE-2004, ACE-2005, and NNE datasets, which verifies the effectiveness of HiTRANS for NNER. Besides, HiTRANS outperforms other baselines on NNE dataset containing 114 categories of entities, which further validates the superiority in recognizing nested entities from complex sentences.

From the tendency, span-based methods and layered-based methods draw more attention than hypergraph-based methods in recent years, which probably because they effectively balance the performance and efficiency. In summary, the overall performance of the HiTRANS demonstrates its superiority in NNER, which benefits from the hierarchical span representation.

4.4 Ablation Study

As shown in Table 3, we present the experimental results of our proposed model on ACE-2005.

[2]We reproduced the results using their implement code (Li et al., 2020), which only obtains 76.18% F1 score, rather than 83.75% F1 score.

Setting	P(%)	R(%)	F1(%)
without CL-EMB	86.28	87.42	86.85
without WL-EMB	84.80	**87.65**	86.20
without SL-EMB	80.46	76.76	78.56
without MHA	85.32	87.32	86.31
without Phase 2	85.86	87.19	86.52
HiTRANS	**86.48**	87.62	**87.04**

Table 3: Ablation study on ACE-2005. MHA denotes the multi-head attention.

The multi-level features (i.e., CL-EMB, WL-EMB, and SL-EMB) obtained from character-level, word-level, and sentence-level are essential for the final performance. Particularly, the sentence-level feature improves the performance by a large margin, which may because the language model usually has a large number of parameters to learn a better representation. Besides, HiTRANS without WL-EMB has a slight increase in recall, but a decrease in precision, which indicates that the word-level feature contributes to select the correct entity from candidate spans. The residual multi-head attention (MHA) contributes to the final performance as well, which could be due to the refined span representations in each layer. In addition, HiTRANS model with two phases shows better performance, which may because phase 2 can further propagate information in a top-down manner. We only remove Phase 2 for ablation studies, since Phase 1 need to take original multi-level representations as input. In all, our HiTRANS achieves 87.04% F1 score, which indicates that all components contribute to the effectiveness and the whole framework has superior in achieving the overall performance.

Sentence	These problems multiplied when the New England chain Stop n' Shop acquired Giant.
Gold Label	New England: [LOC]; the New England chain: [ORG]; the New England chain Stop n' Shop :[ORG]; Giant: [ORG]
Exhaustive	the New England chain Stop n' Shop :[ORG]; New England chain: [ORG]; Giant: [ORG]
Layered	the New England chain Stop n' Shop :[ORG]; Giant: [ORG]
Boundary-aware	New England: [LOC]; Giant: [ORG]; the New England chain Stop n' Shop :[ORG]
Pyramid	the New England chain Stop n' Shop :[ORG]; the New England chain: [ORG]; New England chain: [ORG]; New England: [LOC]; n' Shop :[ORG]; Giant: [ORG];
Our model	Giant: [ORG]; New England: [LOC]; the New England chain: [ORG]; the New England chain Stop n' Shop :[ORG]

Table 4: A case study of the NNER.

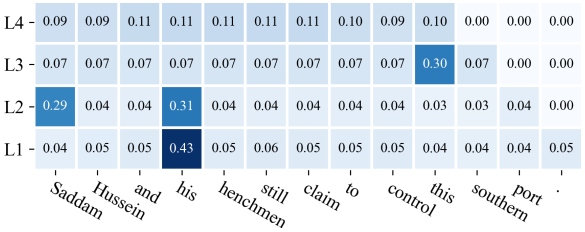

Figure 4: The confidences of entities in different layers.

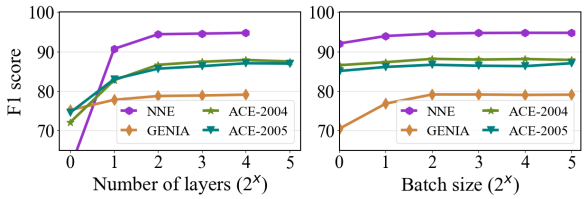

Figure 5: Parameter sensitivity analysis of HiTRANS. The out-of-memory problem occurs when the number of layers is set to 32 (i.e., 2^5) on the NNE and GENIA dataset, as shown at the left.

4.5 Case Study and Visualization

Table 4 shows a case study comparing our model with Exhaustive (Sohrab and Miwa, 2018), Layered (Ju et al., 2018), Boundary-aware (Zheng et al., 2019), and Pyramid (Jue et al., 2020) models, which are more germane and representative. In this example, there is an entity “the New England chain Stop n’ shop” containing the entity “the New England chain", which also has an entity “New England" nested in it. Our proposed model recognizes all potential entities of different-length in a fine-to-coarse manner. Exhaustive gets the wrong token of entity heads and misses the token “the" in entities, and Layered merely extracts outer entities. Compared with the Pyramid model detecting wrong spans, our HiTRANS can extract both inner and outer entities more precisely in a hierarchical manner. It demonstrates that HiTRANS contributes to the performance of NNER, which may due to the hierarchical transformer refines span representations in each layer. Furthermore, the hierarchical label prediction model has the advantage of identifying nested named entities by incorporating both semantic dependencies.

For in-depth analysis of HiTRANS, we visualize the result of the predictions in each layer with masking. Owing to space limit, only the first four layers are shown in Figure 4. From the input sentence, “his” is correctly recognized as entities of 1-token with 0.43 confidence in Layer 1, "Saddam Hussein" and "his Henchmen" are recognized as entities of 2-token with 0.29 and 0.31 confidence in Layer 2, respectively. Likewise, other spans of l-token in Layer $l \in L$ are assigned with different confidences. In a word, we can observe that the recognized entities of different lengths are assigned with higher confidences than others in each layer, which contributes to distill truth named entities from candidate spans and further validates the effectiveness of our HiTRANS for the NNER.

4.6 Parameter Sensitivity Analysis

Two primary parameters, i.e., the number of layers and batch size, are selected to verify the impact of parameters on the effectiveness of HiTRANS. The number of layers denotes how many layers used in the hierarchical model and the batch size controls the size of allocated resources. To study uncertainty in the output of our HiTRANS, we adopt the single-parameter sensitivity analysis by varying one parameter while fixing the others each time. As Figure 5 shows, when the number of layers and the batch size change, especially when the number of layers is greater than 4, and the batch size is greater than 4, HiTRANS still maintains high performance on these four benchmark datasets. Although the number of layers is related to the maximum nesting depth, the results demonstrate that HiTRANS is not sensitive to parameter settings and has superior performance and robustness in NNER.

5 Conclusion

This paper presents a novel HiTRANS framework, which learns effective span representations for label prediction of nested entities in a hierarchical manner. The proposed framework iteratively generates candidate span representations by aggregating adjacent features and further refines them based on a bottom-up and top-down transformer network. Moreover, a candidate span is further recognized as a named entity sequentially, leveraging the semantic dependency of adjacent spans. Extensive experimental results demonstrate that HiTRANS achieves the state-of-the-art performances on GENIA, ACE-2004, ACE-2005 and NNE datasets.

Acknowledgments

This work is partially supported by National Natural Science Foundation of China through grants No.61976102, No.U19A2065, and No.61902145, and HKBU direct grant (Ref. AIS 21-22/02).

References

Billy Chiu, Gamal Crichton, Anna Korhonen, and Sampo Pyysalo. 2016. How to train good word embeddings for biomedical nlp. In *BioNLP*, pages 166–174.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. Bert: Pre-training of deep bidirectional transformers for language understanding. In *NAACL-HLT*, pages 4171–4186.

George R Doddington, Alexis Mitchell, Mark A Przybocki, Lance A Ramshaw, Stephanie M Strassel, and Ralph M Weischedel. 2004. The automatic content extraction (ace) program-tasks, data, and evaluation. In *Lrec*, volume 2, pages 837–840. Lisbon.

Jenny Rose Finkel and Christopher D Manning. 2009. Nested named entity recognition. In *EMNLP*, pages 141–150.

Joseph Fisher and Andreas Vlachos. 2019. Merge and label: A novel neural network architecture for nested ner. In *ACL*, pages 5840–5850.

Meizhi Ju, Makoto Miwa, and Sophia Ananiadou. 2018. A neural layered model for nested named entity recognition. In *NAACL-HLT*, pages 1446–1459.

WANG Jue, Lidan Shou, Ke Chen, and Gang Chen. 2020. Pyramid: A layered model for nested named entity recognition. In *ACL*, pages 5918–5928.

Arzoo Katiyar and Claire Cardie. 2018. Nested named entity recognition revisited. In *NAACL-HLT*, 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(suppl_1):i180–i182.

Zhenzhong Lan, Mingda Chen, Sebastian Goodman, Kevin Gimpel, Piyush Sharma, and Radu Soricut. 2020. ALBERT: A lite BERT for self-supervised learning of language representations. In *ICLR*.

Xiaoya Li, Jingrong Feng, Yuxian Meng, Qinghong Han, Fei Wu, and Jiwei Li. 2020. A unified MRC framework for named entity recognition. In *ACL*, pages 5849–5859.

Hongyu Lin, Yaojie Lu, Xianpei Han, and Le Sun. 2019. Sequence-to-nuggets: Nested entity mention detection via anchor-region networks. In *ACL*, pages 5182–5192.

Wei Lu and Dan Roth. 2015. Joint mention extraction and classification with mention hypergraphs. In *EMNLP*, pages 857–867.

Ying Luo and Hai Zhao. 2020. Bipartite flat-graph network for nested named entity recognition. In *ACL*, pages 6408–6418.

Aldrian Obaja Muis and Wei Lu. 2017. Labeling gaps between words: Recognizing overlapping mentions with mention separators. In *EMNLP*, pages 2608–2618.

Jeffrey Pennington, Richard Socher, and Christopher D Manning. 2014. Glove: Global vectors for word representation. In *EMNLP*, pages 1532–1543.

Nicky Ringland, Xiang Dai, Ben Hachey, Sarvnaz Karimi, Cecile Paris, and James R Curran. 2019. Nne: A dataset for nested named entity recognition in english newswire. In *ACL*, pages 5176–5181.

Takashi Shibuya and Eduard Hovy. 2020. Nested named entity recognition via second-best sequence learning and decoding. *Transactions of the Association for Computational Linguistics*, 8:605–620.

Mohammad Golam Sohrab and Makoto Miwa. 2018. Deep exhaustive model for nested named entity recognition. In *EMNLP*, pages 2843–2849.

Jana Straková, Milan Straka, and Jan Hajic. 2019. Neural architectures for nested ner through linearization. In *ACL*, pages 5326–5331.

Chuanqi Tan, Wei Qiu, Mosha Chen, Rui Wang, and Fei Huang. 2020. Boundary enhanced neural span classification for nested named entity recognition. In *AAAI*, pages 9016–9023.

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 *NIPS*, pages 5998–6008.

C Walker, S Strassel, J Medero, and K Maeda. 2006. Ace 2005 multilingual training corpus. *Progress of Theoretical Physics Supplement*, 110(110):261–276.

Bailin Wang and Wei Lu. 2018. Neural segmental hypergraphs for overlapping mention recognition. In *EMNLP*, pages 204–214.

Bailin Wang, Wei Lu, Yu Wang, and Hongxia Jin. 2018. A neural transition-based model for nested mention recognition. In *EMNLP*, pages 1011–1017.

Congying Xia, Chenwei Zhang, Tao Yang, Yaliang Li, Nan Du, Xian Wu, Wei Fan, Fenglong Ma, and Philip Yu. 2019. Multi-grained named entity recognition. In *ACL*, pages 1430–1440.

Zhiwei Yang, Hechang Chen, Jiawei Zhang, Jing Ma, and Chang Yi. 2020. Attention-based multi-level feature fusion for named entity recognition. In *IJCAI*, pages 3594–3600.

Wonjin Yoon, Chan Ho So, Jinhyuk Lee, and Jaewoo Kang. 2019. Collabonet: collaboration of deep neural networks for biomedical named entity recognition. *BMC bioinformatics*, 20(10):249.

Changmeng Zheng, Yi Cai, Jingyun Xu, Ho-fung Leung, and Guandong Xu. 2019. A boundary-aware neural model for nested named entity recognition. In *EMNLP*, pages 357–366.