Propose-and-Refine: A Two-Stage Set Prediction Network for Nested Named Entity Recognition

Shuhui Wu, **Yongliang Shen**, **Zeqi Tan** and **Weiming Lu***
College of Computer Science and Technology, Zhejiang University
{shwu, syl, zqtan, luwm}@zju.edu.cn

Abstract

Nested named entity recognition (nested NER) is a fundamental task in natural language processing. Various span-based methods have been proposed to detect nested entities with span representations. However, span-based methods do not consider the relationship between a span and other entities or phrases, which is helpful in the NER task. Besides, span-based methods have trouble predicting long entities due to limited span enumeration length. To mitigate these issues, we present the Propose-and-Refine Network (PnRNet), a two-stage set prediction network for nested NER. In the propose stage, we use a span-based predictor to generate some coarse entity predictions as entity proposals. In the refine stage, proposals interact with each other, and richer contextual information is incorporated into the proposal representations. The refined proposal representations are used to re-predict entity boundaries and classes. In this way, errors in coarse proposals can be eliminated, and the boundary prediction is no longer constrained by the span enumeration length limitation. Additionally, we build multi-scale sentence representations, which better model the hierarchical structure of sentences and provide richer contextual information than token-level representations. Experiments show that PnRNet achieves state-of-the-art performance on four nested NER datasets and one flat NER dataset.

1 Introduction

Named Entity Recognition (NER) aims to detect the span and category of all entities in text, which is an essential task in natural language processing. Notably, named entities are often nested in other external entities. For instance, in the sentence "This indeed was one of Uday's homes", the entity "Uday" is nested in the entity "Uday's homes" while "Uday's homes" is also nested in another larger entity "one of Uday's homes". This is because natural language sentences are hierarchical. Smaller-scale entities might be nested in larger-scale entities as sub-constituency trees.

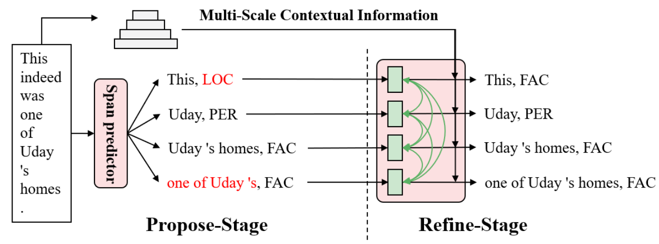

Figure 1: Span-based predictors are error-prone (we color prediction errors in red). The entity "This" is misclassified due to a lack of interaction with other related phrases in span-based predictors. Besides, Span-based methods cannot predict long entities "one of Uday 's homes" if we set a small enumeration length limit. PnRNet addresses these issues with proposal refinement and re-prediction.

Various methods have been proposed to handle the nested NER task, such as optimized sequence-tagging methods [Ju *et al.*, 2018; Straková *et al.*, 2019], hypergraph methods [Lu and Roth, 2015; Katiyar and Cardie, 2018], transition-based methods [Wang *et al.*, 2018]. These methods, however, either require complex manual-designed tagging schemas or suffer from error propagation. Recently, span-based methods, which perform classification over features of candidate spans, have gained popularity and have achieved promising results in the NER task [Sohrab and Miwa, 2018; Tan *et al.*, 2020; Shen *et al.*, 2021; Wang *et al.*, 2020]. Unlike previous methods, span-based prediction can naturally address the nested NER task without complex detecting schemas and does not suffer from error propagation. However, as shown in Figure 1, span-based methods still have the following two issues. First, the prediction of an entity may rely on other phrases in the sentence. But span representations are typically generated through features of tokens that constitute the span. Therefore the relationship between a span and other phrases or entities is not considered in span-based methods, making the span-based methods error-prone. Second, the length of the enumerated span is always limited since exhaustive enumeration is computationally expensive. Therefore it is hard for span-based methods to detect long entities.

This paper presents the Propose-and-Refine Network (PnRNet), a two-stage set prediction network for the nested NER. To address the two previously mentioned issues of the span-based methods, we apply a two-stage decoding procedure

*Corresponding author

to detect named entities, as shown in Figure 1. In the propose stage, we use a span-based predictor to generate a set of coarse entity predictions as proposals. In the refine stage, proposals are fed into the transformer decoder [Vaswani *et al.*, 2017], where proposals interact with each other, and richer contextual information is aggregated into the proposal representations. Finally, the refined proposal representations are used to re-predict entity boundaries and classes. In this way, the prediction errors of the coarse proposals can be eliminated with enriched information, and the boundary prediction is not constrained by the enumeration length limitation of the span-based predictor. The final predictions are considered as a set, and a permutation-invariant loss is applied to train the model.

Additionally, we build multi-scale sentence representations to provide richer contextual information in the decoder. As mentioned before, natural language sentences are hierarchical. Therefore, representing the input sentence as a hierarchical structure is natural and helps solve the nested NER problem. For that purpose, we collect the span features generated in the propose stage to form multi-scale sentence representations. In this way, proposals can directly interact with features of spans highly related to the predicted entity rather than token features in the refine stage, which can aggregate hierarchical contextual information more effectively.

Our main contributions are as follows:

- We present a novel propose-and-refine two-stage set prediction network for the nested NER task. With richer contextualized information aggregated in the refine stage, PnRNet can make more precise predictions than the span-based predictor. Moreover, PnRNet is not constrained by the span enumeration length because we re-predict entity boundaries and classes after proposal refinement.
- To model the hierarchical structure of natural language sentences and better detect nested named entities, we build multi-scale features for decoding to provide richer hierarchical contextual information.
- Experiments on ACE04, ACE05, GENIA, KBP17, and CoNLL03 show that our model outperforms all previous models. We also conduct a detailed ablation study to validate the effectiveness of these innovations.

2 Model

In this section, we will introduce PnRNet, a two-stage set prediction network for nested NER, as illustrated in Figure 2.

2.1 Stage I: Propose

Span Feature Generation. Given an input sentence $\mathbf{X}$ of length N, we concatenate the contextual embedding $\mathbf{x}_i^{\text{plm}}$ generated by a pre-trained language model, word embedding $\mathbf{x}_i^{\text{word}}$, part-of-speech embedding $\mathbf{x}_i^{\text{pos}}$, and character embedding $\mathbf{x}_i^{\text{ch}}$ of each token, and then feed the concatenated embeddings into a BiLSTM [Hochreiter and Schmidhuber, 1997] for token-level representation $\mathbf{x}_i$:

$$\mathbf{x}_i = \text{BiLSTM}([\mathbf{x}_i^{\text{plm}}; \mathbf{x}_i^{\text{word}}; \mathbf{x}_i^{\text{pos}}; \mathbf{x}_i^{\text{ch}}]) \quad (1)$$

where $[;]$ denotes concatenation operation.

We generate span features from token-level sentence representations in a bottom-up manner:

$$\mathbf{h}_{l,i} = \begin{cases} \text{Linear}([\mathbf{h}_{l-1,i}; \mathbf{h}_{l-1,i+1}]) & \text{if } l > 1 \\ \mathbf{x}_i & \text{if } l = 1 \end{cases} \quad (2)$$

where $\mathbf{h}_{l,i}$ denotes the feature of the span (l, i), which is the l-gram span starting from the i-th token. We limit the bottom-up construction process to spans of length L since exhaustive span enumeration is computationally expensive, especially for long sentences.

Entity Proposal. A span-based predictor is used to classify the entity type of each span with the span features generated in the previous step. The classification scores of span (l, i) is computed as follows:

$$\mathbf{p}_{l,i}^{\text{cls}} = \text{Softmax}(\text{Linear}(\mathbf{h}_{l,i})) \quad (3)$$

Then the likelihood of that span being an entity can be obtained by:

$$p_{(l,i)\in\mathcal{E}} = \sum_{\text{ty}_t \neq \varnothing} \mathbf{p}_{l,i}^{\text{cls}}(\text{ty}_t) \quad (4)$$

where $\mathbf{p}_{l,i}^{\text{cls}}(\text{ty}_t)$ indicates the probability of the span (l, i) to be an entity of type ty_t. $\mathcal{E}$ represents all entities in the sentence and $\varnothing$ is a pseudo entity type which means this span is not an entity.

Span features are sorted by $p_{(l,i)\in\mathcal{E}}$ in descending order, and top-K span features which are most likely to be entities will be picked as the entity proposals $\mathbf{Q} \in \mathbb{R}^{K\times d}$.

It is worth noting that in the nested NER task, the prediction of an entity may rely on other related phrases or entities. However, the span-based predictor does not model the relationship between a span and other phrases. Therefore, the span-based predictor is error-prone, and these entity proposals are just coarse predictions. We have to incorporate richer contextual information into the proposal representation in the refine stage to get more precise predictions.

2.2 Stage II: Refine

PnRNet uses a transformer decoder [Vaswani *et al.*, 2017] to refine the coarse entity proposals. The transformer decoder is composed of a stack of M transformer decoder layers. We denote $\mathbf{U}_m \in \mathbb{R}^{K\times d}$ as the output of decoder layer m. The coarse entity proposals are fed into the transformer decoder as the input of the first decoder layer $\mathbf{U}_0 = \mathbf{Q}$. The output of each decoder layer will be fed into the next layer, forming an iterative refining process.

Self-attention. Entities in a sentence are related to each other. Therefore, modeling the relationship between different entities is helpful for NER. In self-attention layer, entity proposals interact with each other through the multi-head attention mechanism:

$$\mathbf{U}_m^{\text{SA}} = \text{MultiHeadAttn}(\mathbf{U}_{m-1}, \mathbf{U}_{m-1}, \mathbf{U}_{m-1}) \quad (5)$$

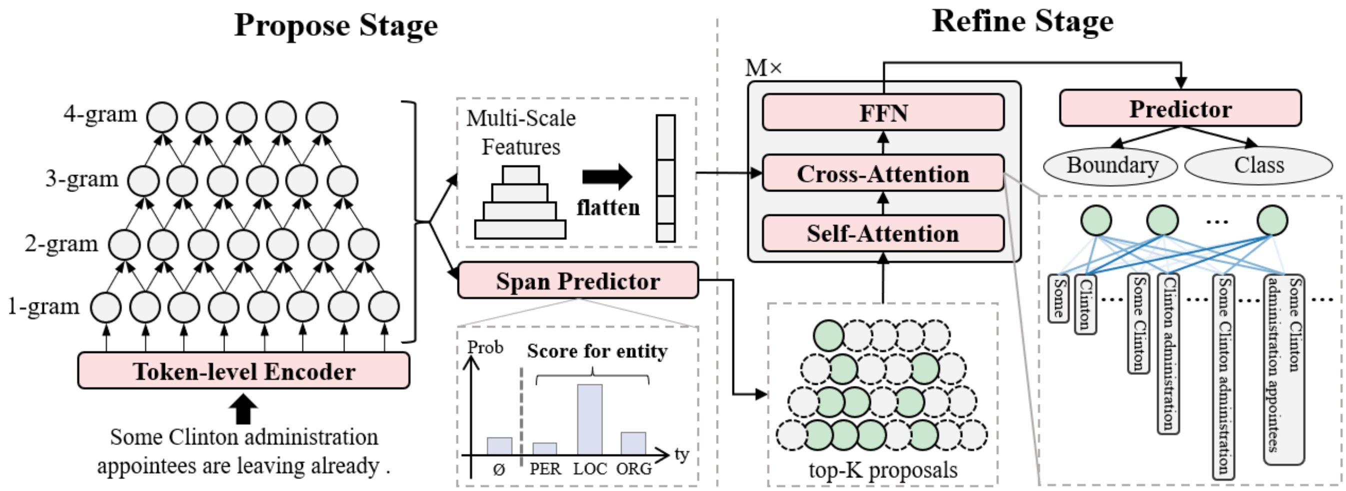

Figure 2: Overview of PnRNet. In the propose stage, PnRNet computes span representations and generates coarse entity proposals with a span-based predictor. In the refine stage, the proposals are refined through a transformer decoder and finally are used to re-predict boundaries and entity classes. We collect multi-scale features from span features generated in the propose stage to provide hierarchical contextual information in proposal refinement. For simplicity of demonstration, we show a PnRNet with span enumeration length limited to $L = 4$.

Cross-attention with multi-scale features. In order to model the relationship between the proposal and other phrases in the input sentences, entity proposals interact with sentence representations through cross-attention so that richer contextual information can be aggregated into the representations of the entity proposals:

$$\mathbf{U}_m^{\text{CA}} = \text{MultiHeadAttn}(\mathbf{U}_m^{\text{SA}}, \mathbf{H}, \mathbf{H}) \tag{6}$$

where $\mathbf{H}$ is sentence representation. Since natural language sentences are hierarchical, we use multi-scale sentence representations to provide hierarchical contextual information for the nested NER task. Therefore, we collect the span representations generated in the propose stage to form layered pyramid-like multi-scale sentence representations:

$$\mathbf{H}_l = [\mathbf{h}_{l,1}, \mathbf{h}_{l,2}, \ldots, \mathbf{h}_{l,N-l+1}] \tag{7a}$$

$$\mathbf{H} = \text{Flatten}([\mathbf{H}_1, \mathbf{H}_2, \ldots, \mathbf{H}_L]) \tag{7b}$$

where $\mathbf{H}_l \in \mathbb{R}^{(N-l+1)\times d}$ is the list of features of spans with length l, $\mathbf{H} \in \mathbb{R}^{c\times d}$ is the list of all span features, and $c = \frac{(2N-L+1)L}{2}$ is the number of the enumerated spans. Since $\mathbf{H}$ contains features of spans of different lengths, $\mathbf{H}$ can be viewed as the multi-scale representation of the input sentence. With multi-scale features, proposal representations can directly attend with features of related spans. Compared with token-level features, using multi-scale features as keys and values in cross-attention can aggregate hierarchical contextual information more effectively.

Feed-forward layer. The entity proposals processed by the self-attention layer and the cross-attention layer will be fed into a feed-forward layer to generate the refined proposals of the current decoder layer:

$$\mathbf{U}_m = \text{Linear}(\text{ReLU}(\text{Linear}(\mathbf{U}_m^{\text{CA}}))) \tag{8}$$

Re-Prediction. In order to eliminate the errors in coarse proposals with the information incorporated in the transformer decoder, we use the output of the last transformer decoder layer ($\mathbf{U}_M$) to re-predict entity classes and boundaries. For each refined proposal $\mathbf{u}_i$ in $\mathbf{U}_M$, we compute the entity classification probability of $\mathbf{u}_i$:

$$\mathbf{p}_i^{\text{cls}} = \text{Softmax}(\text{Linear}(\mathbf{u}_i)) \tag{9}$$

For boundary detection, we first fuse refined entity proposal $\mathbf{u}_i$ with 1-gram span features (token-level features):

$$\mathbf{H}_i^{\text{fuse}} = [[\mathbf{u}_i; \mathbf{h}_{1,1}], [\mathbf{u}_i; \mathbf{h}_{1,2}], \ldots, [\mathbf{u}_i; \mathbf{h}_{1,N}]] \tag{10}$$

And then we perform classification over the fused features to obtain the probability of each token to be the left and right boundary of the entity:

$$\mathbf{p}_i^{\delta} = \text{Softmax}(\text{MLP}_{\delta}(\mathbf{H}_i^{\text{fuse}})) \quad \delta \in \{\text{l}, \text{r}\} \tag{11}$$

where MLP is multi-layer perceptron.

2.3 Training Objective

Proposal Loss. We first calculate the loss of the entity proposals generated in the propose stage. The span-based entity proposal generation is a type classification task, so we use cross-entropy to obtain the loss between ground truth entity type and span-based entity classification of all enumerated spans:

$$\mathcal{L}_{\text{proposal}} = -\sum_{l=1}^{L}\sum_{i=1}^{N-l+1} log\mathbf{p}_{l,i}^{\text{cls}}(c_{l,i}) \tag{12}$$

where $c_{l,i}$ is the ground truth entity type of span (l, i).

Refine Loss. The final entity predictions of our PnRNet are order-agnostic, so we consider them as a set $\hat{y} = \{(\mathbf{p}_k^{\text{cls}}, \mathbf{p}_k^{\text{l}}, \mathbf{p}_k^{\text{r}}) \mid k = 1, \ldots, K\}$. Following Tan *et al.* [2021], we compute a permutation-invariant set loss between the final entity predictions and ground truths. We first define the match cost between gold entity $y_k = (c_k, l_k, r_k)$ and prediction indexed by $\sigma(k)$:

$$\mathcal{L}_{\text{match}}(y_k, \hat{y}_{\sigma(k)}) = -\mathbb{1}_{\{c_k \neq \varnothing\}}[\mathbf{p}_{\sigma(k)}^{\text{cls}}(c_k) + \mathbf{p}_{\sigma(k)}^{\text{l}}(l_k) + \mathbf{p}_{\sigma(k)}^{\text{r}}(r_k)] \tag{13}$$

where $\mathbb{1}$ denotes the indicator function. Then, we find an optimal match between prediction set and gold entities:

$$\hat{\sigma} = \underset{\sigma \in \mathfrak{S}_K}{\arg\min} \sum_k^K \mathcal{L}_{\text{match}}(y_k, \hat{y}_{\sigma(k)}) \tag{14}$$

This optimal assignment problem can be easily solved by the Hungarian algorithm [Kuhn, 1955]. The loss for the refine stage is defined as the sum of the classification loss and the boundary prediction loss of all K predictions:

$$\mathcal{L}_{\text{refine}}(y, \hat{y}) = -\sum_{k=1}^{K}\{\lambda^{\text{cls}} \log \mathbf{p}_{\hat{\sigma}(k)}^{\text{cls}}(c_k) + \lambda^{\text{b}} \mathbb{1}_{\{c_k \neq \varnothing\}}[\log \mathbf{p}_{\hat{\sigma}(k)}^{\text{l}}(l_k) + \log \mathbf{p}_{\hat{\sigma}(k)}^{\text{r}}(r_k)]\} \tag{15}$$

where λ^{cls}, λ^{b} are loss weights. We train the model with auxiliary losses, i.e., using the output of each decoder layer to predict entities and sum losses of all layers up for fast convergence.

3 Experiments

3.1 Setting

Dataset. We conduct experiments on four wildly used nested NER datasets – ACE04 [Doddington *et al.*, 2004], ACE05 [Walker *et al.*, 2006], GENIA [Ji *et al.*, 2017], and KBP17 [Ohta *et al.*, 2002]. Following [Katiyar and Cardie, 2018], we split samples of ACE04 and ACE05 into train, dev, test set by 8:1:1, and split samples of GENIA into train/dev, test set by 9:1. For KBP17, we split all documents into 866/20/167 documents for train, dev, and test set, following [Lin *et al.*, 2019]. We also conduct experiments on a flat NER dataset, CoNLL03 [Tjong Kim Sang and De Meulder, 2003].

Evaluation metric. Entity prediction is considered correct when both span and category are correctly predicted. We consider precision, recall, and F1 score as our evaluation metrics. We additionally report classification F1 score and localization F1 score in the ablation study for detailed analysis.

Implementation details. We use pre-trained BERT [Devlin *et al.*, 2019] as the contextual encoder. For a fair comparison, we use the BERT-base-cased model for the KBP17 dataset, BERT-large-cased model for ACE04, ACE05, and CoNLL03 datasets, and BioBERT-large-cased-v1.1 [Lee *et al.*, 2020] for GENIA dataset. We use GloVe (100d) [Pennington *et al.*, 2014] as our pre-trained word embedding in all experiments

Model	ACE04		
	Pr.	Rec.	F1
Katiyar and Cardie [2018]	73.60	71.80	72.70
Straková *et al.* [2019]	-	-	84.40
Li *et al.* [2020]	85.05	86.32	85.98
Wang *et al.* [2020]	86.08	86.48	86.28
Yu *et al.* [2020]	87.30	86.00	86.70
Yan *et al.* [2021]	87.27	86.41	86.84
Tan *et al.* [2021]	**88.46**	86.10	87.26
Shen *et al.* [2021]	87.44	87.38	87.41
PnRNet	87.90	**88.34**	**88.12**
Model	**ACE05**		
	Pr.	Rec.	F1
Katiyar and Cardie [2018]	70.60	70.40	70.50
Lin *et al.* [2019]	76.20	73.60	74.90
Wang *et al.* [2020]	83.95	85.39	84.66
Yan *et al.* [2021]	83.16	86.38	84.74
Yu *et al.* [2020]	85.20	85.60	85.40
Li *et al.* [2020]	87.16	86.59	86.88
Shen *et al.* [2021]	86.09	87.27	86.67
Tan *et al.* [2021]	**87.48**	86.63	87.05
PnRNet	86.27	**89.04**	**87.63**
Model	**GENIA**		
	Pr.	Rec.	F1
Lin *et al.* [2019]	75.80	73.90	74.80
Straková *et al.* [2019]	-	-	78.31
Wang *et al.* [2020]	79.45	78.94	79.19
Yan *et al.* [2021]	78.87	79.6	79.23
Tan *et al.* [2021]	82.31	78.66	80.44
Yu *et al.* [2020]	81.80	79.30	80.50
Shen *et al.* [2021]	80.19	80.89	80.54
PnRNet	**82.68**	**81.04**	**81.85**
Model	**KBP17**		
	Pr.	Rec.	F1
Ji *et al.* [2017]	76.20	73.00	72.80
Lin *et al.* [2019]	77.70	71.80	74.60
Li *et al.* [2020]	80.97	81.12	80.97
Tan *et al.* [2021]	84.91	83.04	83.96
Shen *et al.* [2021]	85.46	82.67	84.05
PnRNet	**86.51**	**84.06**	**85.27**
Model	**CoNLL03**		
	Pr.	Rec.	F1
Lample *et al.* [2016]	-	-	90.94
Devlin *et al.* [2019]	-	-	92.8
Straková *et al.* [2019]	-	-	93.38
Wang *et al.* [2020]	-	-	93.43
Li *et al.* [2020]	92.33	94.61	93.04
Yu *et al.* [2020]	**93.7**	93.3	93.5
PnRNet	93.18	**94.14**	**93.66**

Table 1: Main Results on four nested NER datasets (ACE04, ACE05, GENIA, KBP17) and one flat NER dataset (CoNLL03). Our PnRNet achieves state-of-the-art performance in F1-score on all these datasets.

Entity Proposal	Proposal Refinement	Multi-Scale Features	ACE04					GENIA				
			Loc. F1	Cls. F1	Pr.	Rec.	F1	Loc. F1	Cls. F1	Pr.	Rec.	F1
✓	✓	✓	**92.34**	**91.75**	**87.90**	**88.34**	**88.12**	**84.65**	88.17	**82.68**	81.04	**81.85**
	✓	✓	89.52	88.40	81.54	85.40	83.42	84.13	87.75	82.58	80.33	81.44
✓			90.35	91.57	84.40	88.04	86.18	84.42	**88.87**	81.10	**82.20**	81.65
✓	✓		91.75	90.78	86.86	87.12	86.99	83.89	87.42	82.02	79.93	80.96

Table 2: Ablation Study

except GENIA and use BioWordVec [Chiu *et al.*, 2016] for the GENIA dataset. We set the span enumeration length limit to $L = 16$, the number of layers of the transformer decoder to $M = 3$. We pick $K = 60$ proposals with the highest scores as entity proposals, a number significantly larger than the number of entities in most sentences.

3.2 Overall Performance

Table 1 demonstrates the overall performance of our PnRNet compared with various baselines. The experiments on nested NER datasets show that our PnRNet outperforms all previous methods by a large margin. Specifically, PnRNet achieves +0.71%, +0.58%, +1.31%, and +1.22% gain in F1-score in ACE04, ACE05, GENIA, and KBP17. On the flat NER dataset CoNLL03, PnRNet also achieves SOTA performance. It shows that modeling interactions between entities and incorporating richer hierarchical contextual information into the entity proposals not only help in detecting nested entities but also improve the performance of the flat NER.

3.3 Ablation Study

We conduct the ablation study in the following three aspects, as shown in Table 2.

Span-based entity proposal. To validate the effectiveness of our proposal generation process, we replace these proposal features with a set of randomly initialized learnable embeddings. The F1-score drops by -4.7% and -0.41% in ACE04 and GENIA datasets without entity proposal. It shows that proposal representations generated in the first stage provide necessary information for entity recognition compared with randomly initialized vectors.

Proposal refinement. In the ablation experiment without proposal refinement, we directly evaluate the performance of the span-based predictor. The performance drops by -1.94% and -0.20% in ACE04 and GENIA compared with full PnRNet. This indicates aggregating richer contextual information and modeling the relationship with other phrases can benefit the performance of NER.

Multi-scale feature. In the ablation experiment without multi-scale features, we use the output of the sequence encoder ($\mathbf{H}' = [\mathbf{x}_1, \mathbf{x}_2, \ldots, \mathbf{x}_N]$), which is the token-level sentence representation, to provide contextual information in proposal refinement. The performance drops by -1.13% and -0.89% in ACE04 and GENIA datasets. It shows that multi-scale sentence representations provide richer hierarchical contextual information, which is helpful in entity recognition.

len	support	Proposal	Final
all	3035	86.18	**88.12**
1	1519 (50.0%)	90.49	**90.54**
2	626 (20.6%)	87.44	**88.14**
3	318 (10.5%)	85.98	**88.06**
4	149 (4.9%)	83.28	**85.62**
5	107 (3.5%)	80.51	**84.65**
6-8	172 (5.7%)	77.01	**83.28**
9-16	105 (3.5%)	62.55	**73.83**
17-	39 (1.3%)	-	**73.42**

Table 3: Comparison of F1-score between the entity proposals generated by the span-based predictor in the propose stage and the final prediction of PnRNet on entities of different lengths in ACE04. The span-based entity predictor cannot propose spans that exceed the span enumeration length limitation L, which is 16 in this experiment.

3.4 Detailed Analysis of the Effect of the Proposal Refinement

For detailed analysis, we compare the performance between span-based proposals and the final predictions of PnRNet on entities of different lengths. As shown in Table 3, as the entity length grows, the performance of the span-based entity recognition (entity proposals) declines significantly. In contrast, the performance of the final prediction only gets a slight drop. Furthermore, the two-stage detection still has satisfactory performance on very long entities, even when it exceeds the prediction limits of the span-based predictor used in the propose stage. This indicates the refine stage of PnRNet, which performs interaction between proposals and incorporates multi-scale contextual information into proposal features, helps a lot in recognizing nested named entities, especially for long entities.

3.5 Visualization of Multi-Scale Cross-Attention Weight

We visualize the cross-attention weight map of the last decoder layer of our PnRNet to confirm the effectiveness of the multi-scale features. As shown in Figure 3, four spans with the highest attention scores are “law professor” (the predicted entity of the proposal), “rick pildes” (the person name of the “law professor”), “law professor rick pildes” (an entity related to “law professor”) and ”you” (another entity mentioned in this sample). This indicates that through multi-scale features, the entity proposal can directly attend to features of spans that are highly related to the proposal in cross-attention.

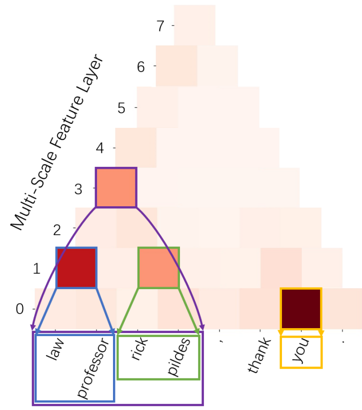

Figure 3: We show an example from the ACE04 dataset to illustrate the multi-scale attention weight of PnRNet. We visualize the cross-attention weight of a certain attention head in the last decoder layer. The query of the illustrated cross-attention weight is an entity proposal that finally predicts “law professor”(PER).

“ Health officials in [[Saudi [Arabia]$_{GPE}$]$_{GPE}$]$_{GPE}$ have asked pilgrims visiting its holy sites to wear masks in crowded places to stop the spread of the MERS coronavirus . ”
On the contrary , the patent owner may continue to pursue [[[its]$_{ORG}$]$_{ORG}$]$_{PER}$ rights through the courts .

Table 4: Case study. We mark coarse proposals in red, the corresponding final predictions in blue, and ground truths in green. For simplicity, we omit other irrelevant proposals and predictions and only show one pair of proposal and its corresponding final prediction in each case.

With the power of multi-scale features, the transformer decoder can aggregate hierarchical information that is helpful to detect named entities, improving the performance of the nested NER.

3.6 Case Study

We illustrate some cases in Table 4 to show that our PnRNet can eliminate errors in coarse proposals through proposal refinement. In case 1, by aggregating richer contextual information, boundary errors in the coarse proposal (“Arabia”) can be fixed in the final prediction (“Saudi Arabia”). In case 2, the pronoun entity “its” is misclassified by the span-based predictor as a `PER`. By interacting between the entity proposal and other proposals and contextual phrases, our PnRNet can correctly classify the entity class as an `ORG`.

4 Related Work

Various methods have been proposed to recognize nested named entities. Since traditional sequence tagging methods [Huang *et al.*, 2015; Lample *et al.*, 2016] which predict a label for each token cannot address nested named entities, some optimized tagging schemes are proposed to cope with the nested NER task [Ju *et al.*, 2018; Straková *et al.*, 2019]. Hypergraph methods [Lu and Roth, 2015; Katiyar and Cardie, 2018] represent the parsing structure of the input sentence as a hypergraph and detect nested entities on the graph. Transition-based methods [Wang *et al.*, 2018] generate a sequence of constituency parsing actions to detect nested entities.

Span-based methods predict entities with span representations. Sohrab and Miwa [2018] exhaustively enumerate spans and generate span representation with boundary token features and pooling features of span tokens. Tan *et al.* [2020] first predict boundary and then perform classification over span features. Wang *et al.* [2020] use a pyramid model to generate span representations layer by layer. Yu *et al.* [2020] use a bi-affine operation to compute span classification scores. Shen *et al.* [2021] perform boundary regression after span-based prediction. Span-based methods can naturally address the nested NER task without complex detecting schemas and have achieved promising performance. However, span representations does not model the relationship with other contextual phrases or entities. Besides, span-based methods have difficulty predicting long entities because the span enumeration length is limited to reduce computational complexity. Our PnRNet solves all these two issues through proposal refinement and re-prediction.

Other studies design new architectures or incorporate different paradigms for the nested NER task. Lin *et al.* [2019] first identify anchor words of entity mentions and then detect entity boundaries. Li *et al.* [2020] use a machine reading comprehension model for the nested NER. Yan *et al.* [2021] model the nested NER as a sequence generation task. Since the nested NER task is essentially an order-agnostic set prediction problem, Tan *et al.* [2021] use a sequence-to-set neural network to detect entities as a set and apply a permutation-invariant set loss for training. However, most of these methods only use token-level encodings as sentence representations, which have difficulty representing the hierarchical structure of natural language sentences. We mitigate this issue with multi-scale sentence representation.

5 Conclusion

This paper presents a novel two-stage set prediction network named Propose-and-Refine Network. Firstly, we use a span-based predictor to generate a set of coarse entity predictions as proposals. Then proposals are fed into a transformer decoder for further refinement and finally re-predict entity boundaries and entity classes. So prediction errors in coarse entity proposals can be eliminated, and the model can better detect long entities. Moreover, we generate multi-scale sentence representations to provide richer hierarchical contextual information of the input sentence. Finally, we apply a cross-entropy loss for the entity proposals and a permutation-invariant set loss for the final predictions. Experiments show that our model achieves state-of-the-art performance on flat and nested NER datasets.

Acknowledgments

This work is supported by the National Key Research and Development Project of China (No. 2018AAA0101900), the Key Research and Development Program of Zhejiang

Province, China (No. 2021C01013), CKCEST, and MOE Engineering Research Center of Digital Library.

References

[Chiu *et al.*, 2016] Billy Chiu, Gamal Crichton, Anna Korhonen, and Sampo Pyysalo. How to Train good Word Embeddings for Biomedical NLP. In *Proceedings of the 15th Workshop on Biomedical Natural Language Processing*, 2016.

[Devlin *et al.*, 2019] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding. In *Proc. of ACL*, 2019.

[Doddington *et al.*, 2004] George R. Doddington, Alexis Mitchell, Mark A. Przybocki, Lance A. Ramshaw, Stephanie M. Strassel, and Ralph M. Weischedel. The Automatic Content Extraction (ace) Program - Tasks, Data, and Evaluation. In *Proc. of Lrec*, 2004.

[Hochreiter and Schmidhuber, 1997] Sepp Hochreiter and Jürgen Schmidhuber. Long Short-Term Memory. *Neural Comput.*, 1997.

[Huang *et al.*, 2015] Zhiheng Huang, Wei Xu, and Kai Yu. Bidirectional LSTM-CRF Models for Sequence Tagging. *ArXiv150801991 Cs*, August 2015.

[Ji *et al.*, 2017] Heng Ji, Xiaoman Pan, Boliang Zhang, Joel Nothman, James Mayfield, Paul McNamee, and Cash Costello. Overview of TAC-KBP2017 13 Languages Entity Discovery and Linking. In *Proc. of Tac*, 2017.

[Ju *et al.*, 2018] Meizhi Ju, Makoto Miwa, and Sophia Ananiadou. A Neural Layered Model for Nested Named Entity Recognition. In *Proc. of ACL*, 2018.

[Katiyar and Cardie, 2018] Arzoo Katiyar and Claire Cardie. Nested Named Entity Recognition Revisited. In *Proc. of ACL*, 2018.

[Kuhn, 1955] H. W. Kuhn. The Hungarian method for the assignment problem. *Naval Research Logistics*, 1955.

[Lample *et al.*, 2016] Guillaume Lample, Miguel Ballesteros, Sandeep Subramanian, Kazuya Kawakami, and Chris Dyer. Neural Architectures for Named Entity Recognition. In *Proc. of ACL*, 2016.

[Lee *et al.*, 2020] Jinhyuk Lee, Wonjin Yoon, Sungdong Kim, Donghyeon Kim, Sunkyu Kim, Chan Ho So, and Jaewoo Kang. BioBERT: A pre-trained biomedical language representation model for biomedical text mining. *Bioinformatics*, 2020.

[Li *et al.*, 2020] Xiaoya Li, Jingrong Feng, Yuxian Meng, Qinghong Han, Fei Wu, and Jiwei Li. A Unified MRC Framework for Named Entity Recognition. In *Proc. of ACL*, 2020.

[Lin *et al.*, 2019] Hongyu Lin, Yaojie Lu, Xianpei Han, and Le Sun. Sequence-to-Nuggets: Nested Entity Mention Detection via Anchor-Region Networks. In *Proc. of ACL*, 2019.

[Lu and Roth, 2015] Wei Lu and Dan Roth. Joint Mention Extraction and Classification with Mention Hypergraphs. In *Proc. of EMNLP*, 2015.

[Ohta *et al.*, 2002] Tomoko Ohta, Yuka Tateisi, and Jin-Dong Kim. The Genia Corpus: An Annotated Research Abstract Corpus in Molecular Biology Domain. In *Proceedings of the Second International Conference on Human Language Technology Research*, 2002.

[Pennington *et al.*, 2014] Jeffrey Pennington, Richard Socher, and Christopher Manning. Glove: Global Vectors for Word Representation. In *Proc. of EMNLP*, 2014.

[Shen *et al.*, 2021] Yongliang Shen, Xinyin Ma, Zeqi Tan, Shuai Zhang, Wen Wang, and Weiming Lu. Locate and Label: A Two-stage Identifier for Nested Named Entity Recognition. In *Proc. of ACL*, 2021.

[Sohrab and Miwa, 2018] Mohammad Golam Sohrab and Makoto Miwa. Deep Exhaustive Model for Nested Named Entity Recognition. In *Proc. of EMNLP*, 2018.

[Straková *et al.*, 2019] Jana Straková, Milan Straka, and Jan Hajic. Neural Architectures for Nested NER through Linearization. In *Proc. of ACL*, 2019.

[Tan *et al.*, 2020] Chuanqi Tan, Wei Qiu, Mosha Chen, Rui Wang, and Fei Huang. Boundary Enhanced Neural Span Classification for Nested Named Entity Recognition. *AAAI*, 2020.

[Tan *et al.*, 2021] Zeqi Tan, Yongliang Shen, Shuai Zhang, Weiming Lu, and Yueting Zhuang. A Sequence-to-Set Network for Nested Named Entity Recognition. In *Proc. of IJCAI*, 2021.

[Tjong Kim Sang and De Meulder, 2003] Erik F. Tjong Kim Sang and Fien De Meulder. Introduction to the CoNLL-2003 Shared Task: Language-Independent Named Entity Recognition. In *Proc. of HLT-NAACL*, 2003.

[Vaswani *et al.*, 2017] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. Attention is All you Need. In *Proc. of NeurIPS*, 2017.

[Walker *et al.*, 2006] Christopher Walker, Stephanie Strassel, Julie Medero, and Kazuaki Maeda. ACE 2005 Multilingual Training Corpus, February 2006.

[Wang *et al.*, 2018] Bailin Wang, Wei Lu, Yu Wang, and Hongxia Jin. A Neural Transition-based Model for Nested Mention Recognition. In *Proc. of EMNLP*, 2018.

[Wang *et al.*, 2020] Jue Wang, Lidan Shou, Ke Chen, and Gang Chen. Pyramid: A Layered Model for Nested Named Entity Recognition. In *Proc. of ACL*, 2020.

[Yan *et al.*, 2021] Hang Yan, Tao Gui, Junqi Dai, Qipeng Guo, Zheng Zhang, and Xipeng Qiu. A Unified Generative Framework for Various NER Subtasks. In *Proc. of ACL*, 2021.

[Yu *et al.*, 2020] Juntao Yu, Bernd Bohnet, and Massimo Poesio. Named Entity Recognition as Dependency Parsing. In *Proc. of ACL*, 2020.