End-to-End Entity Detection with Proposer and Regressor

Xueru Wen · Changjiang Zhou · Haotian Tang · Luguang Liang · Hong Qi · Yu Jiang

Accepted: 16 February 2023 / Published online: 13 April 2023
© The Author(s), under exclusive licence to Springer Science+Business Media, LLC, part of Springer Nature 2023

Abstract

Named entity recognition is a traditional task in natural language processing. In particular, nested entity recognition receives extensive attention for the widespread existence of the nesting scenario. The latest research migrates the well-established paradigm of set prediction in object detection to cope with entity nesting. However, the manual creation of query vectors, which fail to adapt to the rich semantic information in the context, limits these approaches. An end-to-end entity detection approach with proposer and regressor is presented in this paper to tackle the issues. First, the proposer utilizes the feature pyramid network to generate high-quality entity proposals. Then, the regressor refines the proposals for generating the final prediction. The model adopts encoder-only architecture and thus obtains the advantages of the richness of query semantics, high precision of entity localization, and easiness of model training. Moreover, we introduce the novel spatially modulated attention and progressive refinement for further improvement. Extensive experiments demonstrate that our model achieves advanced performance in flat and nested NER, achieving a new state-of-the-art F1 score of 80.74 on the GENIA dataset and 72.38 on the WeiboNER dataset.

Keywords

Named entity recognition · Set prediction · Attention · Feature pyramid
1 Introduction

Named entity recognition identifying text spans of specific entity categories is a fundamental task in natural language processing. It has played a crucial role in many downstream tasks such as relation extraction [1], information retrieval [2] and entity linking [3]. The model [4, 5] based on sequence labeling has achieved great success in this task. Even though mature and efficient as these models are, they fail to handle the nested entities that are a non-negligible scenario in the real-world language environment. Some recent studies [6] have noted the formal consistency of Object detection with NER tasks. Figure 1 shows the instances where the entities overlap with each other and the detection boxes intersect with each other.

A few previous works have designed proprietary structures to deal with the nested entities, such as the constituency graph [9] and hypergraph [10]. Other works [11, 12] capture entities through the layered model containing multiple recognition layers. Despite the success achieved by these approaches, they inevitably necessitate the deployment of sophisticated transformations and costly decoding processes, introducing extra errors compared to the end-to-end manner.

Seq2Seq methods [13] can address various kinds of NER subtasks in a unified form. However, these methods have difficulties defining the order of the outputs due to the natural conflict between sets and sequences. This trait limits the performance of the model. The span-based approaches [14, 15], which identify entities by enumerating all candidate spans in a sentence and classifying them, also receive lots of attention. Although enumeration can be theoretically perfect, the high computational complexity still burdens these methods. Second, these methods mainly focus on learning span representations without the supervision of entity boundaries [16]. Further, enumerating all subsequences from a sentence generate many negative samples, which reduces the recall rate. Some recent work, including set prediction networks, has attempted to address these defects.

The latest works [17] treat information extraction as a reading comprehension task, extracting entities and relations through manually constructed queries. The set prediction network [18] is introduced to the entity and relation extraction. Because the techniques accommodate the unordered character of the prediction target, these methods achieve great success.

Fig. 1 Examples for object detection and named entity recognition under flat and nested circumstances. Examples are obtained from GENIA [7] and COCO2017 [8].
However, most of them still confront problems caused by query vectors. The random initialization of the query vector leads to the lack of sufficient semantic information and difficulty in learning the proper attention pattern.

This paper presents the end-to-end entity detection network, which predicts the entities in a single run and thus is no longer affected by prediction order. The proposed model transforms the NER task into a set prediction problem. First, we utilize the feature pyramid network to build the proposer, which generates high-quality entity proposals with rich semantical query vectors, high-overlapping spans, and category logarithms. High-quality proposals significantly alleviate the difficulty of training. Then, the encoder-only regressor constructed by the iterative transformer employs the regression procedure on the entity proposals. In contrast to some span-based methods discarding the partially match proposals, the regressor adjusts these proposals to improve model performance. The prediction head computes probability distributions for each entity proposal to identify the entities. In the training phase, we dynamically assign prediction targets to each proposal.

Moreover, we introduce the novel spatially modulated attention in this paper. It guides the model to learn more reasonable attention patterns and enhances the sparsity of the attention map by making full use of the spatial prior knowledge, which improves the model’s performance. We also correct the entity proposal at every layer of the regressor network, called the progressive refinement. This strategy increases the precision of the model and facilitates gradient backpropagation.

Our contribution can be summarized as follows:

- We design the proposer constructed by the feature pyramid to incorporate multi-scale features and initialize high-quality proposals with high-overlapping spans and strongly correlated queries. Compared to previous works which randomly initialize query vectors, the proposer network may greatly reduce training difficulty.
- We deploy the encoder-only framework in the regressor, which evades the handcraft construction of query vectors and hardships in learning appropriate query representations and thus notably eases the difficulties of convergence. The iterative refinement strategy is further utilized in the regressor to improve the precision and promote gradient backpropagation.
- We introduce the novel spatially modulated attention mechanisms that help learn the proper attention pattern. The spatially modulated attention mechanism dramatically improves the model’s performance by integrating the spatially prior knowledge to increase attention sparsity.

## 2 Related Work

In this section, we will review some related works about Named entity recognition and set prediction. We will analyze different methodologies for the NER task and the trend of set prediction algorithm development.

### 2.1 Named Entity Recognition

Since traditional NER methods with sequence labeling [19, 20] have been well studied, many works [21] have been devoted to extending sequence tagging methods to nested NER. One of the most explored methods is the layered approach [22]. Other works deploy proprietary structures to handle the nested entities, such as the hypergraph [10]. Although these methods
have achieved advanced performance, they are still not flexible enough due to the need for manually designed labeling schemes. In contrast, the end-to-end framework proposed in this paper avoids this disadvantage and thus facilitates the implementation and migration of the method.

Seq2Seq approach [23] unifies different forms of nested entity problems into sequence generation problems. Even though this strategy avoids the complicated annotation methods, the sensitivity of the decoding order and beam search algorithm poses a barrier to boosting the model performance. The end-to-end model in this paper incorporates the set prediction algorithm in order to overcome the difficulties confronting the Seq2Seq model.

Span-based approaches [24, 25], which classify candidate spans to identify the entities, also draw broad interest. The span-based method formally resembles the object detection task in computer vision. Based on the long-standing idea [26] of associating the image with natural language, some [6] proposes a two-stage identifier that fully exploits partially matched entity proposals and introduces the regression procedure. Despite the instructiveness of the insight to migrate proven methods in computer vision to NER tasks, the error propagation due to two-stage models and the proper way for boundary regression in languages remain issues to address. In this paper, we comprise the propose and regression stage into an end-to-end framework and refine the entity proposal based on probability distribution to fit the language background of the NER task.

2.2 Set Prediction

Several recent pieces of research [18, 27] have deployed the set prediction network in information extraction tasks and proved its effectiveness. These works can be seen as variants of DETR [28] which is proposed for object detection and use the transformer decoder to update the manually created query vector for the generation of detection boxes and corresponding categories.

The models based on set prediction networks, especially DETR, have been extensively studied. Slow convergence due to the random initialization of the object query is the fundamental obstacle of DETR. A two-stage model [29] with the feature pyramid [30] which generates high-quality queries and introduces multi-scale features is proposed to settle the convergence problem. This paper also discusses the necessity for cross-attention and suggests that an encoder-only network can achieve equally satisfactory results. Spatially modulated co-attention [31] integrating spatially prior knowledge is also introduced to ease the problem. This work increases the sparsity of attention through a priori knowledge for the purpose of accelerating training. The thought-provoking deformable attention [32] is presented, which shows the possibility of models learning the spatial structure of the attention. It also improves the model performance by iteratively refining the detection box.

A considerable amount of improvement work has been proposed to solve the convergence of set prediction problems. Previous work [27] employing set prediction requires delicate selection for the number of queries to reach a promising speed of model convergence. Inspired by the analysis [29] done previously, we propose the end-to-end framework to settle the problem.
3 Method

In this section, we are going to detail our method. The general framework of our model is shown in Fig. 2, which is constructed in the following parts:

- **Sentence encoder** We utilize hybrid embedding to encode the sentence. The generated embeddings are then fused by the BiGRU [33] to produce the final multi-granularity representation of the sentence.
- **Proposer** We build up the feature pyramid through the stack of BiGRU and CNN [34] to constitute the proposer network. The proposer exploits the multi-scale features to initialize the entity proposals.
- **Regressor** We design the regressor that refines the proposals progressively to locate and classify spans more accurately. The regressor is built by stacking update layers constructed by the spatially modulated attention mechanism.
- **Prediction head** The prediction head outputs the span location probability distribution based on the refined proposals. The distribution will be combined with probabilities generated by the category logarithms to compute the joint probability distribution, from which we can obtain the eventual prediction results.

3.1 Sentence Encoder

The goal of this component is to transform the original sentence into dense hidden representations. With the inputted sentence \( S = [w_1, w_2, \ldots, w_L] \), we represent the \( i \)-th token \( w_i \) with the concatenation of multi-granularity embeddings as follows:

\[
h'_i = [h_i^{\text{char}}; h_i^{\text{bert}}; h_i^{\text{word}}; h_i^{\text{pos}}]
\]

(1)

The embedding at character level \( h_i^{\text{char}} \) is generated by fusing each character’s embedding \( h^c \) through recurrent neural networks and average pooling them as follows:

\[
h_i^{\text{char}} = \text{Pool} \left( \text{BiGRU} \left( [h_1^{c}, h_2^{c}, \ldots, h_D^{c}] \right) \right)
\]

(2)

where \( D \) is the number of characters constituting the token. The character-level embedding can help the model cope better with out-of-vocabulary words.

\( h_i^{\text{bert}} \) stands for the representation generated by the pre-trained language model BERT [35]. We follow [36] to obtain the contextualized representation by encoding the sentence with the surrounding tokens. The BERT separates the tokens into subtokens by Wordpiece partitioning [37]. The representation of subtokens is average pooled to create the contextualized...
embedding as follows:

\[ h_{i}^{\text{bert}} = \text{Pool} \left( [h_{1}^{b}, h_{2}^{b}, ..., h_{O}^{b}] \right) \]  

(3)

where \( O \) is the number of subtokens forming the token. The pre-trained model can aid in the generation of more contextually relevant text representations.

As for the embedding of word-level \( h_{i}^{\text{word}} \), we exploit pre-trained word vectors including Glove [38]. To introduce the semantic message of part-of-speech, we embed each token’s POS tag as \( h_{i}^{\text{pos}} \).

The multi-granularity embeddings are then fed into the BiGRU network to produce the hybrid embedding for the final representation \( H \) of the sentence as follows:

\[ H' = [h_{1}', h_{2}', ..., h_{L}'] \]

\[ H = W\text{BiGRU} (H') + b \]

(4)

3.2 Proposer

We tailor the pyramid network [22] to build up our proposer, which is able to integrate features of multi-scales and reasonably initialize the proposals. Figure 3 illustrates the structure of the feature pyramid. The constitution of the pyramid is performed in a bottom-to-up and up-to-bottom manner. The bidirectional construction procedures allow better message communication between layers. We selectively merge the features at different layers to yield initial proposals. This process is implemented in a similar way to the attention mechanism.

3.2.1 Forward Block

The feature pyramid is first built from the bottom to up. It consists of \( L \) layers, each with two main components, a BiGRU and a CNN of kernel sizes \( k \). At layer \( l \), the BiGRU models the interconnections of spans of the same size. The CNN aggregates \( k \) neighboring hidden states, which are then passed into higher layers. Apparently, each feature vector represents a span of \( v_{l} \) original tokens and \( v_{l} \) can be calculated as:

\[ v_{l} = 1 - l + \sum_{i=1}^{l} k_{i} \]  

(5)
One may note that the pyramid structure provides inherent induction: the higher the number of layers, the shorter the input sequence, with higher levels of feature vectors representing the long entities and lower levels representing the short entities. Moreover, since the input scales of the layers are diverse, we apply Layer Normalization [39] before feeding the hidden states into the BiGRU.

As described above, the forward block can be formalized as follow:

\[
\begin{align*}
H'_l &= WBiGRU \left( LayerNorm \left( H'_l \right) \right) + b \\
H'_{l+1} &= \sigma \left( Conv1D \left( H_l \right) \right)
\end{align*}
\]

where \( \sigma \) is activate function which we exploit the Gelu [40] in our work. In particular, there is \( H'_0 = [h_0, h_1, h_2, ..., h_L] \), namely the initial inputs to the pyramid network is exactly the output from the sentence encoder.

### 3.2.2 Backward Block

The backward block allows layers to receive feedback from the higher-level neighbors, which benefits the model to learn better representations at each layer by enhancing the communication between the neighbor tokens. In bottom-to-up propagation, the sequence reduces the length each time passes through the CNN. Thus the corresponding Transposed CNNs are required to reconstruct the representation of the text span embeddings.

Specifically, in the \( l-1 \)-th inverse layer, we first feed the hidden states transmitted down from the \( l \)-th layer into BiGRU to catch the spans’ interaction. And then, we concatenate the outputs with the outcomes from the \( l-1 \)-th forward layer. The concatenated representations are linear transformed and then summed up with the pooled residuals to get the outcomes at the \( l-1 \)-th layer. The outcomes will be passed through the Transposed CNN and then sent to the next layer. Similarly, Layer Normalization is deployed due to the diversity of input scales.

The overall structure can be formalized as follows:

\[
\begin{align*}
H'_{l-1} &= H'_l \oplus BiGRU \left( LayerNorm \left( H^b_{l} \right) \right) \\
H_{l-1} &= LayerNorm \left( Pool \left( H \right) + WH'_l + b \right) \\
H^b_{l-1} &= \sigma \left( TransposedConv1D \left( H_{l-1} \right) \right)
\end{align*}
\]

where \( \sigma \) is activate function and Gelu is utilized. \( \oplus \) denotes the concatenation operation. Here, the length of the residuals and the feature representations of the current layer were aligned using average pooling. The purpose of designing this residual connection is to avoid model degradation with the increase of the feature pyramid layers.

### 3.2.3 Proposal Fuse

After the build-up of the feature pyramid, we are able to initialize the entity proposals. In contrast to the works [18, 27] which manually create the query vectors, the encoder-only architecture is deployed in our work in which query vectors also work as semantic feature vectors.

Note that the entity proposal mentioned in this paper includes not only query vector \( q_i \in \mathbb{R}^d \) but also category logarithms \( c_i \in \mathbb{R}^{T+1} \) and span location \( n_i \in \mathbb{R}^2 \), where \( d \) stands for the dimension of queries vector and \( T \) represents the number of the pre-defined categories. Additional one for \( T+1 \) corresponds to the None type. The span location indicates the start
and end position of the entity. The proposals are initialized with the help of the prepared feature pyramid.

Apparently, each feature vector $h_{i,l}$ in the pyramid specifically corresponds to a text span $n_{i,l} = [s_{i,l}, e_{i,l}]$. Our model computes the scores vector $r_{i,l}$ for each feature and utilizes them to calculate the weights vector for each proposal as follows:

$$r_{i,l} = MLP (h_{i,l})$$

$$= [r^s_{i,l}, r^e_{i,l}]$$

$$\alpha^j_{i,l} = \frac{\exp (r_{i,l})}{\sum_{(i',l') \in N_j} \exp (r_{i',l'})}$$

where $\alpha^j_{i,l} \in [0, 1]^2$ denotes the weight vector of the span at index $i$ of the $l$-th level for $j$-th proposal that corresponds to the token $w_j$. $N_j = \{(i, l) | j \in [s_{i,l}, e_{i,l}]\}$ present positions of features whose corresponding text span contains the token $w_j$. The coefficients of aggregation at the start and end positions are calculated separately, which can enhance the expressiveness of the aggregation process and increase the range of the initial span.

With the weight vector, the location span $n_j$ can be initialized as follows:

$$n_j = \sum_{(i,l) \in N_j} \alpha^j_{i,l} \cdot p_{i,l}$$

where $\cdot$ stands for the element-wise product and $p_{i,l}$ represent the span location corresponding to the feature $h_{i,l}$. One may notice that the position of the initial span is actually bounded by the leftmost span and the rightmost span containing the token $j$. Such limitation helps the model to give more reasonable proposals.

As for query vector $q_j$, we directly employ the features at the bottom layer, namely $h_{i,1}$. The reason for doing so is that the features at the bottom layer are obtained from the whole propagation, which means they contain the most semantic messages of the pyramid. Once obtain the query vector, the category logarithms $c_j$ can be naturally derived using a multi-layer perceptron:

$$c_j = MLP (q_j)$$

The category logarithms will be further utilized and corrected in the regressor network.

### 3.3 Regressor

Previous works [29, 41] have proven the importance of the sparsity of the attention map in accelerating the training. Based on their conclusion, we introduce the spatially modulated attention in the regressor. We employ the encoder-only architecture in which semantic feature vectors directly play the role of the query instead of manually creating them. Inspired by successful practice on auxiliary loss [28, 42], we further introduce progressive refinement into the regressor. The overall architecture of the regressor is shown in Fig. 4.

#### 3.3.1 Category Embedding

Each proposal is inputted alongside category logarithms, which carry the semantic information of the proposal category. We follow the way of embedding the positional message in the
Transformer [43] using the element-wise sum to integrate the query vector with the category information:

\[ \alpha_i = \text{Softmax} (c_i) \]
\[ H^c = [h^c_1, h^c_2, ..., h^c_{T+1}] \]
\[ h'_i = q_i + \alpha_i^T H^c \] (11)

where \( \alpha_i \) stands for the category weights vector of \( i \)-th proposal. \( H^c \in \mathbb{R}^{d \times (T+1)} \) represents the type embedding matrix. \( h'_i \) denotes the category embedded query vectors.

In an encoder-only design, query vectors are also semantic feature vectors, making it difficult for the model to discriminate the role played by each query vector. We created category embeddings based on this understanding to clarify the function of each query vector in the forward propagation process.

### 3.3.2 Spatially Modulated Attention

The central idea of the spatially modulated attention mechanism is to utilize the spatial guidance from the span locations of each entity proposal to learn the reasonable sparse attention maps.

**Dynamic spatial weight maps** A Gaussian-like distribution is generated for each proposal in each attention head. The Gaussian-like distribution in this paper is given as:

\[ G(x) = e^{(x-\mu)^T \Theta (x-\mu)} \] (12)

where \( x \in \mathbb{R}^2 \) denotes independent variable. \( \mu \in \mathbb{R}^2 \) can be seen as the expectation vector, which represents the center of the distribution. \( \Theta \in \mathbb{R}^{2 \times 2} \) can be seen as the precision matrix (namely the inverse of the covariance matrix \( \Sigma^{-1} \)), which described the shape of the distribution.

The main difference between the Gaussian-like distribution and the multidimensional Gaussian distribution is that we remove the probability normalization term. The main reason for doing so is twofold. First, the normalized coefficients of the multidimensional Gaussian distribution are derived in the continuous case. Whereas the case here is a discretized attention map. Second, the subsequent Softmax operation will also do the normalization work.
Sets of parameters are needed to generate the distributions. We formulate our ways of producing them as follows:

\begin{equation}
\Delta n_{i,m}, \theta_{i,m} = MLP \left(h^q_{i,m} \right) \\
\mu_{i,m} = n_i + \Delta n_{i,m} \\
\Theta_{i,m} = \theta_{i,m} \theta^T_{i,m}
\end{equation}

where \(h^q_{i,m}\) is the linear transformation for \(m\)-th head of \(h^i_i\) in Formula 11, which is also the query in the QKV Attention\(^{[43]}\). \(\Delta n_{i,m}\) denotes the difference between the span location of the proposal and the center of the map. From Formula 13, it can be found that \(\Theta_{i,m}\) is a symmetric positive definite matrix satisfying the requirement of being the precision matrix for a Gaussian distribution.

It can be observed that the model generates the spatial weight map dynamically, implying that we expect the model to learn the way to make better use of the spatial prior. **Spatially modulated attention** The spatially modulated attention mechanism aims to ease the difficulty of learning the proper attention pattern by enhancing the sparsity of the attention map with the spatial weight map. With the dynamically generated spatial prior \(G\), the spatially modulated attention can be conducted as follows:

\begin{equation}
r^m_{i,j} = \left( W^e_m h^i_i + b^e_m \right)^T \left( W^v_m h^v_{j} + b^v_m \right) + \log G_{i,m}(n_j)
\end{equation}

\begin{equation}
\alpha^m_{i,j} = \frac{\exp \left(r^m_{i,j} \right)}{\sum_{j'=1}^{L} \exp \left(r^m_{i,j'} \right)}
\end{equation}

\begin{equation}
o_{i,m} = \sum_{j=1}^{L} \alpha^m_{i,j} \left( W^v_m h^v_{j} + b^v_m \right)
\end{equation}

where \(m\) denotes the \(m\)-th attention head, \(M\) stands for the number of heads. The spatially modulated attention mechanism performs the element-wise sum between the spatial graph and the dot product attention and then undertakes the Softmax normalization. By introducing the spatial graph, each query vector has a relatively high weight for the query vector whose corresponding span location is closer. It limits the search space for attention patterns and thus reduces the difficulty of the convergence.

**Multi-head iteration** We aggregate the results of each attention head to produce new query vectors and span locations. This process is similar to the way the proposer fuses the proposals. First, we compute the weight vectors for each head:

\begin{equation}
r_{i,m} = MLP \left(o_{i,m} \right) \\
= \left[r^s_{i,m}, r^e_{i,m} \right]
\end{equation}

\begin{equation}
o_{i,m} = \frac{\exp \left(r_{i,m} \right)}{\sum_{m'=1}^{M} \exp \left(r_{i,m'} \right)} \\
= \left[o^s_{i,m'}, o^e_{i,m'} \right]
\end{equation}

where \(M\) is the number of attention heads, \(r_{i,m}\) denotes the score of \(m\)-th head in \(i\)-th proposal. We calculate the corresponding score based on the aggregation result of each head and do the Softmax normalization to get the weight vectors.

Using the weight vectors, we conduct the iteration of the span location. Similarly, we perform aggregation for the start and end positions separately. The procedure can be formulated
as follows:

\[ n_i = \sum_{m=1}^{M} \alpha_{i,m} \cdot \mu_{i,m} \]  

(16)

The spatially modulated attention implements iteration of span location based on the weighted average of each head’s center. Fusing iterations on span location and multi-head attention mechanisms increases the flexibility of span updates and benefits the accuracy of model predictions.

As for query vectors, we simply follow the custom way of aggregating the output of each head as follows:

\[ q_i' = \sum_{m=1}^{M} (W_m o_{i,m} + b_m) \]  

(17)

### 3.3.3 Gated Update

We replace the feed-forward module in Transformer with the gate mechanism in the GRU [44] to strengthen the expressive ability of the model. The gate mechanism can be formulated as follows:

\[
\begin{align*}
    r &= \text{Sigmoid} \left( W_{xr} x + b_{xr} + W_{hr} h + b_{hr} \right) \\
    z &= \text{Sigmoid} \left( W_{xz} x + b_{xz} + W_{hz} c + b_{hz} \right) \\
    n &= \text{Tanh} \left( W_{xn} x + b_{xn} + r \odot (W_{hn} h + b_{hn}) \right) \\
    o &= (1 - z) \odot n + z \odot h
\end{align*}
\]  

(18)

where \( x \) is the current input, \( h \) is the hidden state, and \( o \) is the final new state output.

We denote the above mechanism as \( \text{Gate}(\cdot) \). The new query vectors are obtained as follows:

\[ q_i = \text{Gate}(h_i', q_i') \]  

(19)

Where \( q_i' \) stands for the outputs from the spatially modulated attention in Formula 17 and \( h_i' \) is category embedded represent of the query in Formula 11. The attention mechanism’s outcome is used as the input to update the query representations in each iteration, with the original state representations serving as the hidden states.

### 3.3.4 Logarithms Iteration

After updating the query vector \( q_i \) and span locations \( n_i \), we further iterative the category logarithms \( c_i \). Same as in span location, the module predicts the difference of logarithms rather than directly outputting the result. The updating procedure can be formulated as follows:

\[
\begin{align*}
    \Delta c_i &= \text{MLP}(q_i) \\
    c_i &= c_i + \Delta c_i
\end{align*}
\]  

(20)

The assumption that each entity proposal may change responsibilities during the iteration is the reason for undertaking logarithms iterations.
3.4 Prediction Head

The prediction head produces predictions based on the joint probability distribution indicated by entity proposals. First, we can use Softmax normalization to convert the category logarithms into the probability distribution:

\[
\Delta c_i = \text{MLP} (q_i) \\
p_i^c = \text{Softmax} (c_i + \Delta c_i)
\]  

(21)

As for span location distribution, it is natural to extend the mechanism of spatially modulated attention as the joint pointer network, which is similar to the fusion of the smooth boundary [45] and the biaffine decoder [46]. Specifically, we calculate the joint probability distribution of the span location in the following way:

\[
g_{i}(s, e) = \left( W_{s} q_{i} + b_{s} \right)^{T} \left( W_{h} s + b_{h} \right) + \left( W_{e} q_{i} + b_{e} \right)^{T} \left( W_{e} e + b_{e} \right) + (W_{e} q_{i} + b_{e})
\]

\[
r_{i}(s, e) = \log G_{i}(s, e) + g_{i}(s, e) / \sqrt{d}
\]

\[
p_i^n = \text{Softmax} \left( \{ r_i(s, e) \mid (s \leq e) \land (s, e \leq L) \} \right)
\]

(22)

where \( G \) is created in the same way as spatially modulated attention. \( g_{i} \) is the score derived from the joint pointer network, which is essentially the sum of two pointer networks.

3.4.1 Training

In the training phase, the determination of the best match for each proposal is a one-to-many Linear Assignment Problem. We formulate the search procedure as follows:

\[
\hat{\beta} = \arg \min_{\beta \in \mathcal{O}_L} \sum_{i=1}^{L} \mathcal{L}_{\text{match}} \left( \hat{\mathcal{Y}}_{\beta(i)}, \mathcal{Y}'_{i} \right)
\]

(23)

where \( \hat{\mathcal{Y}} = \mathcal{Y} \cup \{ (\emptyset, \emptyset, \emptyset) \} \) denotes the targets with padding triple denoting None. \( \mathcal{Y} = \{ (s_i, e_i, t_i) \}_{i=1}^{N} \) denotes the set of entities existing in the sentence. \( \mathcal{Y}' \) signifies the proposals outputted by the model, in which each element can be represented as \((p^c, p^n)\). The padding in \( \hat{\mathcal{Y}} \) is necessary as there is the possibility is no entity in the sentence, in which case all the proposals should target at None. The \( \mathcal{L}_{\text{match}} \) is calculated in the following way:

\[
\mathcal{L}_{\text{match}} \left( \hat{\mathcal{Y}}_{\beta(i)}, \mathcal{Y}'_{i} \right) = -\log p_{i}^{c} (t_{\beta(i)}) - \mathbb{I}_{t_{\beta(i)} \neq \emptyset} \left[ \log p_{i}^{n} (s_{\beta(i)}, e_{\beta(i)}) \right]
\]

(24)

\( \mathcal{O}_L \) in Formula 23 stands for the assignment mappings satisfying certain constraints when the number of proposals is \( L \). For the circumstance when \( L > N \), any \( \beta \in \mathcal{O}_L \) has to be surjective mapping, which means every candidate entity will be assigned to a proposal. As for \( L \leq N \), the mapping \( \beta \) ensure \( \hat{\mathcal{Y}}_{\beta(i)} \in \mathcal{Y} \), namely any proposal will be assign to an entity. Given the constraints and optimization objectives, we use Algorithm 1 to solve the assignment problem.

In the algorithm, \( C \) stands for the cost matrix recording the matching cost of each proposal-entity pair. So the \( C_{i,j} \) denotes the matching cost between \( \mathcal{Y}'_i \) and \( \hat{\mathcal{Y}}_j \) and \( C_i \) represents a vector in which element is the cost between \( \mathcal{Y}'_i \) and a target in \( \hat{\mathcal{Y}} \). \textsc{HUG} denotes the Hungarian algorithm [47], which can solve the assignment problem given \( L \leq N \). We use the greedy strategy when \( L > N \), which means the entity with the minimal matching loss (including padding) for each unassigned proposal in the Hungarian algorithm will be the corresponding prediction target.

\( \copyright \) Springer
Algorithm 1 Assignment

Require: Targets $\hat{Y}$, Proposals $Y'$
1: for $i = 1$ to $L$ do
2:     for $j = 1$ to $N$ do
3:         $C_{i,j} \leftarrow L_{\text{match}} (\hat{Y}_j, Y'_i)$
4:     end for
5: end for
6: $\hat{\beta} \leftarrow \text{HUG}(C)$
7: for $i$ in $\{ j | Y'_j $ haven’t been assigned in $\hat{\beta} \}$ do
8:     $k \leftarrow \text{argmin}(C_i)$
9:     $\hat{\beta} \leftarrow \hat{\beta} \cup \{ Y'_i \rightarrow \hat{Y}_k \}$
10: end for

With the optimal pairing $\hat{\beta}$, the final bipartite loss function is simply the matching cost:

$$\mathcal{L} (\hat{Y}, Y') = \sum_{i=1}^{L} L_{\text{match}} (\hat{Y}_{\hat{\beta}(i)}, Y'_i)$$ (25)

3.4.2 Predict

In the prediction phase, we utilize the probabilities distribution outputted by the model to produce the final results through Algorithm 2.

Algorithm 2 Prediction

Require: Proposals $Y'$
1: $E \leftarrow \{\}$
2: for $(p^c, p^n)$ in $Y'$ do
3:     $t \leftarrow \text{argmax}(p^c)$
4:     $(s, e) \leftarrow \text{argmax}(p^n)$
5:     if $t \neq \emptyset \land p^c(t) \cdot p^l(s, e) \geq p^c(\emptyset)$ then
6:         $E \leftarrow E \cup (s, e, t)$
7: end if
8: end for

We examine each proposal to see if it is more likely to represent the entity than to be the padding. In particular, we assume that the probability that a proposal represents an entity is the product of its corresponding category probability and the localization probability, and the probability that it is the padding is the probability that its category is None. So a proposal is considered to signify an entity when and only when the maximum of its span location probability times the maximum of its category probability overpasses its probability of being None.

4 Experiments

The datasets, baselines, and settings used in our experiments are described in this section. We perform a series of experiments to illustrate the characters of our model. Table 1 displays the statistics of the datasets.
Table 1  Statistical information on English datasets

| Info            | GENIA                  | CoNLL03                | WeiboNER               |
|-----------------|------------------------|------------------------|------------------------|
|                 | Training    | Test      | Dev       | Training    | Test      | Dev       | Training    | Test      | Dev       |
| #S              | 15203       | 1669      | 1854      | 14041       | 3250      | 3453      | 1350       | 270       | 270       |
| #AS             | 25.4        | 24.6      | 26.0      | 14.5        | 15.8      | 13.4      | 33.6       | 33.2      | 33.8      |
| #NE             | 46142       | 4367      | 5506      | 23499       | 5942      | 5648      | 1834       | 371       | 405       |
| #AE             | 1.94        | 2.14      | 2.08      | 1.43        | 1.43      | 1.42      | 1.24       | 1.19      | 1.20      |
| #ME             | 21          | 14        | 25        | 20          | 31        | 20        | 17         | 13        | 12        |

#S: the number of sentences. #AS: the average length of sentences. #NE: the number of entities. #AE: the average length of entities. #ME: the maximum number of entities in a sentence

4.1 Datasets

Experiments were carried out using public datasets including GENIA [7], CoNLL03 [48], and WeiboNER [49].

GENIA is a biological dataset including five entity types: DNA, RNA, protein, cell lineage, and cell type. It contains a high portion of nested entities. The same experimental setup as [46] was used. We did not conduct experiments on additional authoritative nested datasets in English because we lacked access to the datasets’ copyright.

CoNLL03 dataset is a huge dataset frequently utilized. The main content in the dataset is the news report from the RCV1 corpus of Reuters. Locations, organizations, people, and information are among the named features. The setting [23] combining the training and validation sets is followed. We demonstrate the generalizability of our approach on flat datasets through experiments on this dataset.

The WeiboNER dataset is a Chinese dataset from the Weibo social platform. It includes named and nominal entity types of individuals, organizations, locations, and geopolitics. The setup of our experiments on this dataset is identical to [50]. We use this Chinese dataset to evaluate the model’s cross-language ability.

4.2 Training Details

We use the pre-trained BERT model developed by the open framework Transformers [51] in our experiments. In order to acquire a better contextual representation in biological domain text, we substituted BERT with BioBERT [52] for the GENIA dataset and used the BIO-word2vec [53] to decrease the quantity of out-of-vocabulary words. We use Glove [38] as pre-trained word vectors for the English dataset. We use word vectors developed by [54] for the Chinese dataset. The models are trained by the AdamW optimizer with the learning rate of $2e^{-5}$. The max gradient normalization is $1e0$ in all experiments. The learning rate is altered along with the training procedure under the cousin warm-up-decay learning rate schedule.

4.3 Comparison

We compare our model to many powerful state-of-the-art models for the GENIA dataset, including LocateAndLabel [6], BARTNER [23], SequenceToSet [27], NER-DP [46], LUKE
Table 2 The performance on the GENIA dataset

| Model                     | GENIA Prec | GENIA Rec | GENIA F1 |
|---------------------------|------------|-----------|----------|
| LocateAndLabel [6]        | 80.19      | 80.89     | 80.54    |
| Pyramid [22]              | 79.45      | 78.94     | 79.19    |
| BARTNER [23]              | 78.89      | 79.60     | 79.23    |
| SequenceToSet [27]        | 82.31      | 78.66     | 80.44    |
| NER-DP [46]               | 81.80      | 79.30     | 80.50    |
| BioBART [57]              | –          | –         | 79.93    |
| LogSumExpDecoder [58]     | 79.20      | 78.67     | 78.93    |
| Our Model                 | 81.74      | 79.76     | 80.74    |

The best results in each column of the table are shown in bold.

Table 3 The performance on CoNLL03 dataset

| Model                     | CoNLL03 Prec | CoNLL03 Rec | CoNLL03 F1 |
|---------------------------|--------------|-------------|------------|
| LocateAndLabel [6]        | 92.13        | 93.79       | 92.94      |
| PIQN [17]                 | 93.29        | 92.46       | 92.87      |
| NER-DP [46]               | 92.85        | 92.15       | 92.50      |
| LUKE [55]                 | –            | –           | 92.87      |
| MRC [56]                  | 92.47        | 93.27       | 92.87      |
| KNN-NER [59]              | 92.82        | 92.99       | 92.93      |
| Our Model                 | 92.86        | 93.13       | 93.00      |

The best results in each column of the table are shown in bold.

The above baselines’ reported results are taken straight from the original published literature.

Table 2 shows the advanced outcomes provided by our model. Our model increases the F1 score compared with the top algorithm in the GENIA dataset. The GENIA dataset contains numerous nested entities, where about 20% of entities are nested. The improvement of our model on the GENIA dataset illustrates our model’s ability to extract nested entities. Compared to the previous state-of-art model LocateAndLabel, we avoid the enumeration of candidate spans through the employment of the set prediction network and achieve the performance. The set prediction can be seen as the soft enumeration with the flexibility lacking in the manual preparation of the candidate spans. Compared with another work SequenceToSet, which also achieved excellent results, we deploy the encoder-only architecture to implement the performance advancement while bypassing the difficulty of training query vectors. The encoder-only architecture accelerates the coverage of the training. It is shown in the following experiments that the model produces comparable results even if only trained under a small number of epochs.

For the CoNLL03 dataset, the LocateAndLabel [6], PIQN [17], NER-DP [46], LUKE [55], MRC [56], and KNN-NER [59] are compared with our model. The results are taken directly from the original published literature. Note that [23] provides the results of LUKE, NER-DP, and MRC.

Table 3 shows the outcomes. Since CoNLL03 is a well-established dataset widely investigated, the improvement on this dataset demonstrates that our model is able to generalize well on the traditional flat dataset. Although our model introduces a certain amount of invalid
query vectors, the satisfactory performance on this dataset shows that the additional entity proposals do not affect the model’s ability to recognize flat entities in general. Compared with some recent works PIQN and KNN-NER, our model still achieves some improvement in F1.

As for the WeiboNER dataset, we compare our model to various strong baselines, including TFM [4], LocateAndLabel [6], KNN-NER [59], SLK-NER [60], BoundaryDet [61], ChineseBERT [62] and AESINER [63]. We present outcomes published in the original literature.

Table 4 shows that our approach reaches a new state-of-art performance provided on the WeiboNER dataset. It demonstrates that the advantages of our model do not limit only to the English datasets. Even compared with some approaches, like ChineseBERT, which targets the Chinese dataset, our model still produces significant improvement. More importantly, since the WeiboNER dataset is a few-shot dataset, the success of this dataset further demonstrates the easiness of training our model. We claim that this excellent performance stems from the good nature of our model’s emphasis on localization, which is generated by spatially modulated attention. Unlike other datasets, the WeiboNER dataset is sampled from social platforms, where the occurrence of entities and the overall sentence meaning are relatively less closely related. Moreover, the entity length tends to be shorter due to the nature of the Chinese language, as seen in Table 1. Spatially modulated attention in the model introduces the Gaussian-like distribution to increase the sparsity of the attention graph by focusing more on local information, which caters well to the characteristics of the data set and thus achieves excellent results.

### 4.4 Detailed Result

As indicated in Table 5, we also looked at the performance of entities of various lengths. Our model produces the best results when the entity length is 1 or 2 on all datasets. And the F1 score gradually decreases as the length of the entity increases.

It is under expectation because short entities can be detected immediately by the proposer, but long entities need the regressor to perform many regressions to identify. The sensitivity of the model’s performance to the entity length is a side consequence of our suggested method’s efficacy. The effect of entity length on recognition performance is also directly influenced by the different kernel sizes of the proposer. From Formula 8 and 9, it can be seen that kernel sizes directly affect the expected length of the model output entities. The expected length in
End-to-End Entity Detection with...

Table 5 Result on entities of different lengths

| Lenz. | GENIA | CoNLL03 | WeiboNER |
|-------|-------|---------|----------|
|       | Prec  | Rec     | F1       | Prec  | Rec     | F1       | Prec  | Rec     | F1       |
| l = 1 | 86.59 | 80.46   | 83.41    | 93.55 | 92.68   | 93.11    | 80.52 | 74.61   | 77.46    |
| l = 2 | 83.01 | 80.19   | 81.58    | **94.04** | 94.32 | **94.18** | 57.31 | 61.84   | 59.49    |
| l = 3 | 77.57 | 78.07   | 77.82    | 90.04 | 91.22   | 90.63    | –     | –       | –        |
| l = 4 | 75.49 | 78.40   | 76.92    | 92.10 | **94.59** | 93.33    | –     | –       | –        |
| l ≥ 5 | 72.99 | 78.99   | 75.87    | 85.18 | 92.00   | 88.46    | –     | –       | –        |
| All   | 81.74 | 79.76   | 80.74    | 92.86 | 93.13   | 93.00    | 72.93 | 71.85   | 72.38    |

The best results in each column of the table are shown in bold.

Table 6 Ablation study

| Model          | GENIA | CoNLL03 | WeiboNER |
|----------------|-------|---------|----------|
|                | Prec  | Rec     | F1       | Prec  | Rec     | F1       | Prec  | Rec     | F1       |
| Origin         | 81.50 | 79.07   | 80.27    | 92.48 | 93.00   | 92.74    | 72.93 | 71.85   | 72.38    |
| -Backward block| 80.58 | 79.65   | 80.11    | 91.72 | 92.66   | 92.19    | 71.42 | 70.37   | 70.89    |
| -Spatial modulation | 81.46 | 78.71   | 80.06    | 92.00 | 92.95   | 92.47    | 71.50 | 69.38   | 70.42    |
| -Gated Update  | 80.85 | 78.78   | 79.80    | 92.36 | 93.11   | 92.73    | 71.18 | 71.35   | 71.27    |
| -Category embedding | 81.39 | 78.58   | 79.96    | 92.20 | 92.79   | 92.49    | 69.64 | 67.40   | 68.50    |
| -Locations iteration | 81.62 | 78.67   | 80.12    | 92.02 | 93.18   | 92.60    | 71.92 | 70.86   | 71.39    |
| -Logarithms iteration | 81.07 | 79.11   | 80.08    | 92.05 | 92.66   | 92.35    | 73.76 | 70.12   | 71.89    |

* '-' means remove the module or substitute it with another module

experiments mostly does not exceed 2, and thus the results in the table are obtained quite reasonably.

4.5 Analysis and Discussion

In this part, we will examine how each module and its corresponding settings affect the algorithm as well as the potential causes.

4.5.1 Ablation Study

On the GENIA dataset, we examined how different modules in the model contributed. Firstly, we deleted the backward module from the feature pyramid to see how it affected communication across features of different layers. Secondly, we tested the importance of spatially prior information by removing spatial modulation from the attention mechanism. Thirdly, we remove the gated update to explore the importance of the gate mechanism in enhancing model expressive ability. Moreover, we remove the category embedding in order to investigate its significance in clarifying the role of each query vector. Finally, we investigated the impact of logarithm and location iterations on the model’s efficacy. The results are shown in Table 6.

The reduction in performance after removing the backward block demonstrates its use in integrating the features of different scales. In addition, since we use the bottom-level features of the feature pyramid as the query vectors used in the subsequent modules, removing the
Table 7  Impact of kernel sizes

| Kernel sizes | GENIA Prec | GENIA Rec | GENIA F1 | CoNLL03 Prec | CoNLL03 Rec | CoNLL03 F1 | WeiboNER Prec | WeiboNER Rec | WeiboNER F1 |
|--------------|------------|-----------|---------|-------------|-------------|-----------|--------------|--------------|-----------|
| [1]          | 81.35      | 78.60     | 79.95   | 92.21       | 92.95       | 92.58     | 73.35        | 71.35        | 72.34     |
| [2]          | 80.88      | 78.44     | 79.69   | 92.34       | 92.88       | 92.61     | 72.44        | 70.12        | 71.26     |
| [2,2]        | 81.50      | 79.07     | 80.27   | 92.48       | 93.00       | 92.74     | 72.58        | 68.64        | 70.55     |
| [2,2,2]      | 81.23      | 78.55     | 79.87   | 91.49       | 92.77       | 92.13     | 72.93        | 71.85        | **72.38** |
| [2,2,2,2]    | 81.24      | 78.67     | 79.94   | 92.24       | 92.63       | 92.43     | 72.19        | 69.87        | 71.01     |
| [2,3]        | 80.09      | **79.09** | 80.01   | 92.00       | 92.74       | 92.37     | 72.68        | 71.60        | 72.13     |
| [2,3,2]      | 81.19      | 78.35     | 79.74   | 91.75       | 92.45       | 92.10     | 71.89        | 71.35        | 71.62     |

The best results in each column of the table are shown in bold. [2, 3] represents a three-layer feature pyramid with two convolution kernels of size 2 and 3. Other symbols indicate structures in the same way. In particular, [] indicates that there is no convolution kernel in the proposer module.

backward module reduces the depth of the network to some extent and weakens the ability of the model to express semantics.

The decrease in F1 scores after taking out spatial modulation emphasizes the need for guidance of spatial prior knowledge. Spatial modulation contributes to the sparsity of the attention map and facilitates the learning of proper attention patterns. It can be observed that the performance drop after removing spatial modulation on the WeiboNER dataset is far more significant than that coming on the other datasets, which is closely related to the nature of the dataset that focus more on local messages.

The reduction of the F1 score after removing the gate mechanism proves it helps improve the model’s expressive ability. It actually plays the same role as the feed-forward layer in the Transformer, i.e., it increases the capacity and nonlinearity of the model.

The performance deterioration after removing category embedding demonstrates the importance of explaining the role of each query vector. Because of the encoder-only architecture used in the proposed approach, the query vector also functions as a semantic vector. This architecture has the potential to introduce confusion about the roles of the query vectors, which is well mitigated by category embedding.

The F1 value reduces when logarithm and location iterations are removed. It emphasizes the significance of iteration in the regression network. We claim it plays a similar role to auxiliary loss [28] and box refinement [64]. In the auxiliary loss, the output results of each layer are decoded and the loss is calculated, thus enabling supervised learning for each layer. In the box refinement, the detection box is gradually adjusted until the eventual prediction is made, which is similar to the iterative process in this paper. Both of them apply the direct updates gradient to each layer, while the latter does the prediction based on the combination of outputs from all layers. These strategies can reduce the difficulty of model training on the one hand. On the other hand, it improves the performance of the model.

### 4.5.2 Impact of Kernel Sizes

We explore the effect of kernel sizes in the pyramid and report the results in Table 7. We tested different layers of the pyramid and various combinations of kernel sizes. All models are trained with 5 epochs, with the number of heads and layers equal to 8 and 3.
The model reaches satisfactory performance when kernel sizes equal \([2, 2]\) or \([2, 2, 2]\). In our opinion, this is due to the fact that under these settings’ entities of various lengths are mostly covered, which allows query vectors to integrate the semantic information of tokens surrounding the proposal for more precise localization. Additionally, the length of the proposals under these kernel settings is close to the average length of entities in the dataset, which diminishes the difficulties of model training.

In addition, we observe that the model outperforms its kernel size of \([2, 2, 2, 2]\) even when the kernel size is \([]\). We believe that the cause of this phenomenon is twofold. On the one hand, different kernel sizes lead to different expected lengths of proposals, and the model will achieve better performance when the expected length of proposals is close to the average entity length. When the kernel size is \([]\), there is no convolution kernel in the proposer module, which means all the proposals will be length 1. So, the expected length of the proposals will be 1, which is closer to the average length of entities in the dataset than when the kernel size is \([2, 2, 2, 2]\). On the other hand, with a kernel size of \([2, 2, 2, 2]\), the feature pyramid has more layers, and the network is deep than when the kernel size is \([]\), which can lead to a relatively more difficult training of the model.

4.5.3 Influence of Regressor Layers

To investigate the influence of the layer, we experimented with models of different layers on the GENIA dataset. All the models were trained with 5 epochs. The sizes of kernels and number of heads equal \([2, 3]\) and 8 relatively. Figure 5 shows the results on different entities length for the models of the different number of layers. We can see our model reaches a comparable result when \(l = 3\). To balance the time cost and model performance, we set \(l = 3\) for other experiments.

It can be directly observed that the recognition efficiency of short entities is less affected by the number of regression layers. On the other hand, the recognition efficiency of entities is more sensitive to the number of regression layers. It is because the proposer can make almost accurate proposals directly for shorter entities, while for long entities, the refinement by the regressor is necessary to identify them correctly.
4.5.4 Analysis of Attention Heads

We also evaluate the model on the GENIA dataset to see how attention heads affect the results. All of the models were trained using 5 epochs with sizes of kernels and the number of layers equal \([2, 3]\) and 3 relatively. The association between the number of heads and model performance is demonstrated in Fig. 6. Our model produces the best results when the number of heads equals 8. So we set it equal to 8 for other experiments.

Similarly, the longer the length of the model, the more the performance of recognizing the entity is affected by the number of attention heads. It is because the number of attention heads directly affects the iterative process of the regressor, which is indispensable for long entities.

4.5.5 Computational Complexity

Theoretically, the number of intermediate proposals generated in the proposer is \(O(kL)\), where \(L\) stands for the length of the sentence and \(k\) stands for the number of layers in the proposer. The amount of outputted high-quality proposals is always identical to the sentence length \(L\). Compared to the traditional span-based methods classifying all the spans, which is of complexity \(O(L^2)\), our proposer is relatively efficient. While for the regressor, our model requires the computation of attention between proposals, which causes the complexity of \(O(nL^2)\). The \(n\) denotes the number of layers in the regressor. On the other hand, if we employ the spatial weights as attention weights instead of fusing them with dot product weights, the theoretical complex will go down to the \(O(nL)\). Empirically, the computation of the proposer and regressor with a proper implementation is highly parallel and much lighter than the sentence encoder with the pre-trained model. So, the extra constituents in the framework should not be the main cause of the efficiency deterioration.

4.5.6 Case Study

We do the case study in Table 8 to show the ability of our model to identify entities in various
Table 8 Case study

| Sentences with Entities | Predictions |
|-------------------------|-------------|
| Treatment of $^2$-T cells $^{11}_{CEL}$ with the selective $^7_{PKC}^{1}_{PRO}$ inhibitor GF109203X abrogates the PMA-induced $^{14}_{IkB\alpha}^{15}_{PRO}$ phosphorylation/degradation irrespective of activation of Ca(2+)-dependent pathways, but not the phosphorylation and degradation of $^{13}_{IkB\alpha}^{14}_{PRO}$ induced by TNF-alpha, a $^{40}_{PKC}^{40}_{PRO}$-independent stimulus | ✓ (2, 3, CEL) ← (2.46, 3.06) ← (2.87, 3.01) ✓ (7, 7, PRO) ← (7.00, 6.83) ← (6.98, 7.01) ✓ (14, 15, PRO) ← (14.41, 15.06) ← (14.76, 15.03) ✓ (33, 34, PRO) ← (33.45, 34.04) ← (33.88, 34.01) ✓ (37, 37, PRO) ← (36.92, 36.90) ← (36.96, 37.01) ✓ (40, 40, PRO) ← (39.93, 39.88) ← (39.97, 40.01) |
| Costimulation with $^2$ anti- $^3_{CD28}$ $^3_{PRO}$ MoAb $^4_{PRO}$ greatly enhanced the proliferative response of $^{11}_{neonatal T}^{12}_{CEL}$ to levels equivalent to those of $^{20}_{adult T}^{21}_{CEL}$ whereas $^{25}_{adult T}^{22}_{CEL}$ only showed slight increases | ✓ (3, 3, PRO) ← (3.12, 2.74) ← (2.98, 3.01) ✓ (2, 4, PRO) ◎ (3.37, 3.96) ← (3.82, 4.02) × (11, 13, DNA) ← (11.95, 13.11) ← (12.61, 13.03) × (12, 13, DNA) ← (12.69, 13.91) ← (13.11, 14.06) × (20, 22, DNA) ← (21.24, 22.15) ← (21.74, 22.02) × (25, 27, DNA) ← (26.03, 27.14) ← (23.14, 24.02) |
| Point mutations of either the $^5_{PU.1}^{15}_{PRO}$ site $^6_{DNA}$ or the $^9_{PU.1}^{19}_{PRO}$ C/EBP $^9_{PRO}$ site $^{10}_{DNA}$ that abolish the binding of the respective factors result in a significant decrease of $^{25}_{GM-CSF}^{25}_{PRO}$ receptor $^{27}_{PRO}$ alpha promoter activity in $^{31}_{myelomonocytic cells}^{32}_{CEL}$ only | ✓ (5, 5, PRO) ← (5.08, 4.77) ← (4.98, 5.01) × (5, 6, CEL) ← (5.45, 6.00) ← (5.86, 6.01) × (9, 10, CEL) ← (9.46, 9.98) ← (9.89, 10.01) ✓ (25, 25, PRO) ← (24.91, 24.83) ← (24.97, 25.02) ✓ (25, 27, PRO) ← (26.17, 27.02) ← (26.73, 27.03) × (31, 32, PRO) ← (31.38, 32.12) ← (31.88, 32.01) |

In the left column, the category of the entity is indicated by the label at the bottom right of the right square bracket and the location of the left and right boundary words is shown by the superscript of the square bracket. The iterative process of the entity proposals, as well as the relationship between them and the predicted entities, are shown in the right column.

cases. The errors demonstrated in the table are also analyzed.

As can be seen in the first example, our model is capable of recognizing multiple entities presented in long sentences. As shown in the table, the proposer is able to produce high-quality proposals. And it can be observed from the iterative process of entity suggestions that the regressor regression plays a crucial role in pinpointing entities even if they are relatively short.

Several significant flaws are demonstrated in the second and third cases. The first problem is that our model has difficulty classifying the entities. While the model correctly locates the entities in all of its predictions, it makes errors in predicting the type of entities. We claim
that this issue arises as a result of the model’s overemphasis on local information. However, the determination of the entity type usually depends on the global message.

The second problem is that there are some circumstances where the model fails to identify highly overlapped nested entities of relatively short length. It is understandable because there are considerably more negative data in the training phase than positive ones in this scenario. Only one proposal represents the genuine entity in most instances where proposals have substantial overlap, while the remaining proposals do not correspond to any entity. It is partly due to the assignment methodology of the entity proposals and the encoder-only architecture.

5 Conclusion and Future Work

An end-to-end entity detection approach with proposer and regressor is presented in this study. We employ a proposer that incorporates multi-scale information through the feature pyramid to predict high-quality entity proposals. It significantly speeds up the training process and boosts performance. The encoder-only framework is proposed in our work, which introduces significant spatially prior knowledge into the attention mechanism. It avoids the unfavorable impact of random initialization of query vectors. In order to increase prediction accuracy, we explore iterations over span locations and category logarithms in the joint model. We model the entity proposal and prediction target assignment issue as a Linear Assignment Problem and compute the bipartite loss during the training phase. The model is able to predict the entities in a single run. The experiments on datasets of different characteristics demonstrate the nature of our model and the effectiveness of our approach. Our work reveals the potential to integrate spatial priors into NLP research. We expect the findings will contribute to a better understanding of the set prediction network and iterative refinement. We will go deep into the relevance of NLP and CV tasks in the future.

Acknowledgements This work was supported by the National Natural Science Foundation of China under Grant Nos. 62072211, 51939003 and U20A20285.

Code Availability The code is available at https://github.com/Rosenberg37/EntityDetection.

Declarations

Conflict of interest The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

References

1. Wang Q, Zhang Q, Zuo M, He S, Zhang B (2022) A entity relation extraction model with enhanced position attention in food domain. Neural Process Lett 54(2):1449–1464. https://doi.org/10.1007/s11063-021-10690-9
2. Zhang L, Zhang S, Balog K (2019) Table2vec: Neural word and entity embeddings for table population and retrieval. In: Proceedings of the 42nd international ACM SIGIR conference on research and development in information retrieval. SIGIR’19. Association for Computing Machinery, New York, NY, pp 1029–1032. https://doi.org/10.1145/3331184.3331333
3. Chen H, Wei B, Liu Y, Li Y, Yu J, Zhu W (2018) Bilinear joint learning of word and entity embeddings for entity linking. Neurocomputing 294:12–18. https://doi.org/10.1016/j.neucom.2017.11.064
4. Liu H, Song J, Peng W, Sun J, Xin X (2022) TFM: a triple fusion module for integrating lexicon information in Chinese named entity recognition. Neural Process Lett. https://doi.org/10.1007/s11063-022-10768-y
5. Yan R, Jiang X, Dang D (2021) Named entity recognition by using XLNet-BiLSTM-CRF. Neural Process Lett 53(5):3339–3356. https://doi.org/10.1007/s11063-021-10547-1

6. Shen Y, Ma X, Tan Z, Zhang S, Wang W, Lu W (2021) Locate and label: A two-stage identifier for nested named entity recognition. In: Proceedings of the 59th annual meeting of the association for computational linguistics and the 11th international joint conference on natural language processing, vol 1, (Long Papers). Association for Computational Linguistics, Online, pp 2782–2794. https://doi.org/10.18653/v1/2021.acl-long.216

7. Ohta T, Tateisi Y, Kim JD (2002) The genia corpus: An annotated research abstract corpus in molecular biology domain. In: Proceedings of the second international conference on human language technology research. HLT ’02. Morgan Kaufmann Publishers Inc., San Francisco, CA, pp 82–86

8. Lin T-Y, Maire M, Belongie S, Hays J, Perona P, Ramanan D, Dollár P, Zitnick CL (2014) Microsoft coco: common objects in context. In: Fleet D, Pajdla T, Schiele B, Tuytelaars T (eds) Computer vision - ECCV 2014. Springer, Cham, pp 740–755

9. Finkel JR, Manning CD (2009) Nested named entity recognition. In: Proceedings of the 2009 conference on empirical methods in natural language processing. Association for Computational Linguistics, Singapore, pp 141–150. https://aclanthology.org/D09-1015

10. Huang H, Lei M, Feng C (2021) Hypergraph network model for nested entity mention recognition. Neurocomputing 423:200–206. https://doi.org/10.1016/j.neucom.2020.09.077

11. Alex B, Haddow B, Grover C (2007) Recognising nested named entities in biomedical text. In: Biological, translational, and clinical language processing. Association for Computational Linguistics, Prague, pp 65–72. https://aclanthology.org/W07-1009

12. Fisher J, Vlachos A (2019) Merge and label: a novel neural network architecture for nested NER. In: Proceedings of the 57th annual meeting of the association for computational linguistics. Association for Computational Linguistics, Florence, pp 5840–5850. https://doi.org/10.18653/v1/P19-1585. https://aclanthology.org/P19-1585

13. Ju M, Miwa M, Ananiadou S (2018) A neural layered model for nested named entity recognition. In: Proceedings of the 2018 conference of the North American chapter of the association for computational linguistics: human language technologies, vol 1 (Long Papers). Association for Computational Linguistics, New Orleans, Louisiana, pp 1446–1459. https://doi.org/10.18653/v1/N18-1131. https://aclanthology.org/N18-1131

14. Xu M, Jiang H, Watcharawittayakul S (2017) A local detection approach for named entity recognition and mention detection. In: Proceedings of the 55th annual meeting of the association for computational linguistics, vol 1 (Long Papers). Association for Computational Linguistics, Vancouver, pp 1237–1247. https://doi.org/10.18653/v1/P17-1114. https://aclanthology.org/P17-1114

15. Sohrab MG, Miwa M (2018) Deep exhaustive model for nested named entity recognition. In: Proceedings of the 2018 conference on empirical methods in natural language processing. Association for Computational Linguistics, Brussels, pp 2843–2849. https://doi.org/10.18653/v1/D18-1309. https://aclanthology.org/D18-1309

16. Tan C, Qiu W, Chen M, Wang R, Huang F (2020) Boundary enhanced neural span classification for nested named entity recognition. In: AAAI. https://doi.org/10.1609/aaai.v34i05.6434

17. Shen Y, Wang X, Tan Z, Xu G, Xie P, Huang F, Lu W, Zhuang Y (2022) Parallel instance query network for named entity recognition. In: Proceedings of the 60th annual meeting of the association for computational linguistics. Association for Computational Linguistics. arXiv:2203.10545

18. Sui D, Chen Y, Liu K, Zhao J, Zeng X, Liu S (2020) Joint entity and relation extraction with set prediction networks. Computation and Language. arXiv:2011.01675

19. Liu M, Tu Z, Zhang T, Su T, Xu X, Wang Z (2022) LTP: a new active learning strategy for CRF-based named entity recognition. Neuro Process Lett. https://doi.org/10.1007/s11063-021-10737-x

20. Wang W, Bao F, Gao G (2019) Learning morpheme representation for Mongolian named entity recognition. Neural Process Lett 50(3):2647–2664. https://doi.org/10.1007/s11063-019-10044-6

21. Metke-Jimenez A, Karimi S (2016) Concept identification and normalisation for adverse drug event discovery in medical forums. In: Proceedings of the first international workshop on biomedical data integration and discovery (BMDID 2016). http://ceur-ws.org/V ol-1709/BMDID_2016_paper_5.pdf

22. Wang J, Shou L, Chen K, Chen G (2020) Pyramid: A layered model for nested named entity recognition. In: Proceedings of the 58th annual meeting of the association for computational linguistics. Association for Computational Linguistics, Online, pp 5918–5928. https://doi.org/10.18653/v1/2020.acl-main.525. https://aclanthology.org/2020.acl-main.525

23. Yan H, Gui T, Dai J, Guo Q, Zhang Z, Qiu X (2021) A unified generative framework for various NER subtasks. In: Proceedings of the 59th annual meeting of the association for computational linguistics and the 11th international joint conference on natural language processing, vol 1, (Long Papers). Association
24. Tan C, Qiu W, Chen M, Wang R, Huang F (2020) Boundary enhanced neural span classification for nested named entity recognition. In: National conference on artificial intelligence. https://doi.org/10.1609/aaai.v34i05.6434
25. Li F, Wang Z, Hui SC, Liao L, Zhu X, Huang H (2021) A segment enhanced span-based model for nested named entity recognition. Neurocomputing 465:26–37. https://doi.org/10.1016/j.neucom.2021.08.094
26. Manning C (2016) Connecting images and natural language a dissertation submitted to the department of computer science and the committee on graduate studies of Stanford university in partial fulfillment of the requirements for the degree of doctor of philosophy
27. Tan Z, Shen Y, Zhang S, Lu W, Zhuang Y (2021) A sequence-to-set network for nested named entity recognition. In: Zhou. -H. (ed.) Proceedings of the thirtieth international joint conference on artificial intelligence, IJCAI-21. International Joint Conferences on Artificial Intelligence Organization, pp 3936–3942. https://doi.org/10.24963/ijcai.2021.542. Main Track
28. Carion N, Massa F, Synnaeve G, Usunier N, Kirillov A, Zagoruyko S (2020) End-to-end object detection with transformers. In: Computer vision-ECCV 2020: 16th European conference, Glasgow, UK, August 23-28, 2020, proceedings, Part I. Springer, Berlin, Heidelberg, pp 213–229. https://doi.org/10.1007/978-3-030-58452-8_13
29. Sun Z, Cao S, Yang Y, Kitani KM (2020) Rethinking transformer-based set prediction for object detection. arXiv: Computer Vision and Pattern Recognition. https://doi.org/10.48550/arXiv.2011.10881
30. Lin TY, Dollár P, Girshick R, He K, Hariharan B, Belongie S (2016) Feature pyramid networks for object detection. arXiv: Computer Vision and Pattern Recognition. https://doi.org/10.48550/arXiv.1612.03144
31. Gao P, Zheng M, Wang X, Dai J, Li H (2021) Fast convergence of detr with spatially modulated co-attention. In: 2021 IEEE/CVF international conference on computer vision (ICCV), pp 3601–3610. https://doi.org/10.1109/ICCV48922.2021.00360
32. Zhu X, Su W, Lu L, Li B, Wang X, Dai J (2021) Deformable detr: Deformable transformers for end-to-end object detection. In: International conference on learning representations. https://doi.org/10.48550/arXiv.2010.04159
33. Cho K, Van Merriënboer B, Gulcehre C, Bahdanau D, Bougares F, Schwenk H, Bengio Y (2014) Learning phrase representations using rnn encoder-decoder for statistical machine translation. arXiv preprint arXiv:1406.1078
34. Kim Y (2014) Convolutional neural networks for sentence classification. In: Proceedings of the 2014 conference on empirical methods in natural language processing (EMNLP). Association for Computational Linguistics, Doha, Qatar, pp 1746–1751. https://doi.org/10.3115/v1/D14-1181. https://aclanthology.org/D14-1181
35. Devlin J, Chang MW, Lee K, Toutanova K (2019) BERT: Pre-training of deep bidirectional transformers for language understanding. In: Proceedings of the 2019 conference of the North American chapter of the association for computational linguistics: human language technologies, vol 1, (Long and Short Papers). Association for Computational Linguistics, Minneapolis, Minnesota, pp 4171–4186. https://doi.org/10.18653/v1/N19-1423. https://aclanthology.org/N19-1423
36. Wang Y, Li Y, Tong H, Zhu Z (2020) HIT: Nested named entity recognition via head-tail pair and token interaction. In: Proceedings of the 2020 conference on empirical methods in natural language processing (EMNLP). Association for Computational Linguistics, Online, pp 6027–6036. https://doi.org/10.18653/v1/2020.emnlp-main.486
37. Wu Y, Schuster M, Chen Z, Le QV, Norouzi M, Macherey W, Krikun M, Cao Y, Gao Q, Macherey K, et al. (2016) Google’s neural machine translation system: bridging the gap between human and machine translation. arXiv preprint arXiv:1609.08144. https://doi.org/10.48550/arXiv.1609.08144
38. Pennington J, Socher R, Manning C (2014) GloVe: Global vectors for word representation. In: Proceedings of the 2014 conference on empirical methods in natural language processing (EMNLP). Association for Computational Linguistics, Doha, Qatar, pp 1532–1543. https://doi.org/10.3115/v1/D14-1162. https://aclanthology.org/D14-1162
39. Ba J, Kiros JR, Hinton GE (2016) Layer normalization. arXiv: Machine Learning. https://doi.org/10.48550/arXiv.1607.06450
40. Hendrycks D, Gimpel K (2016) Gaussian error linear units (gelus). arXiv preprint arXiv:1606.08415. https://doi.org/10.48550/arXiv.1606.08415
41. Jiang Z, Yu W, Zhou D, Chen Y, Feng J, Yan S (2020) Convertb: improving bert with span-based dynamic convolution. In: Neural information processing systems. https://doi.org/10.48550/arXiv.2008.02496
42. Al-Rfou R, Choe D, Constant N, Guo M, Jones L (2018) Character-level language modeling with deeper self-attention. arXiv: Computation and Language. https://doi.org/10.48550/arXiv.1808.04444
43. Vaswani A, Shazeer N, Parmar N, Uszkoreit J, Jones L, Gomez AN, Kaiser L, Polosukhin I (2017) Attention is all you need. arXiv . https://doi.org/10.48550/arXiv.1706.03762
44. Cho K, van Merriënboer B, Gulcehre C, Bahdanau D, Bougares F, Schwenk H, Bengio Y (2014) Learning phrase representations using RNN encoder–decoder for statistical machine translation. In: Proceedings of the 2014 conference on empirical methods in natural language processing (EMNLP). Association for Computational Linguistics, Doha, Qatar, pp 1724–1734. https://doi.org/10.3115/v1/D14-1179. https://aclanthology.org/D14-1179
45. Zhu E, Li J (2022) Boundary smoothing for named entity recognition. https://doi.org/10.48550/arXiv.2204.12031
46. Yu J, Bohnet B, Poesio M (2020) Named entity recognition as dependency parsing. In: Proceedings of the 58th annual meeting of the association for computational linguistics. Association for Computational Linguistics, Online, pp 6470–6476. https://doi.org/10.18653/v1/2020.acl-main.577. https://aclanthology.org/2020.acl-main.577
47. Kuhn HW (1955) The Hungarian method for the assignment problem. Naval Res Logist Q 2:83–97. https://doi.org/10.1002/nav.3800020109
48. Tjong Kim Sang EF, De Meulder F (2003) Introduction to the CoNLL-2003 shared task: language-independent named entity recognition. In: Proceedings of the seventh conference on natural language learning at HLT-NAACL 2003, pp. 142–147. https://aclanthology.org/W03-0419
49. Peng N, Dredze M (2015) Named entity recognition for Chinese social media with jointly trained embeddings. In: Proceedings of the 2015 conference on empirical methods in natural language processing. Association for Computational Linguistics, Lisbon, Portugal, pp 548–554. https://doi.org/10.18653/v1/D15-1064. https://aclanthology.org/D15-1064
50. Li X, Yan H, Qiu X, Huang X (2020) FLAT: Chinese NER using flat-lattice transformer. In: Proceedings of the 58th annual meeting of the association for computational linguistics. Association for Computational Linguistics, Online, pp 6836–6842. https://doi.org/10.18653/v1/2020.acl-main.611. https://aclanthology.org/2020.acl-main.611
51. Wolf T, Debut L, Sanh V, Chaumond J, Delangue C, Moi A, Cistac P, Rault T, Louf R, Funtowicz M, Brew J (2019) Huggingface’s transformers: state-of-the-art natural language processing. arXiv: Computation and Language . https://doi.org/10.48550/arXiv.1910.03771
52. Lee J, Yoon W, Kim S, Kim D, So CH, Kang J (2019) BioBERT: a pre-trained biomedical language representation model for biomedical text mining. Bioinformatics 36(4):1234–1240. https://doi.org/10.1093/bioinformatics/btz682
53. Chiu B, Crichton G, Korhonen A, Pyysalo S (2016) How to train good word embeddings for biomedical NLP. In: Proceedings of the 15th workshop on biomedical natural language processing. Association for Computational Linguistics, Berlin, Germany, pp 166–174. https://doi.org/10.18653/v1/W16-2922. https://aclanthology.org/W16-2922
54. Li S, Zhao Z, Hu R, Li W, Liu T, Du X (2018) Analogical reasoning on Chinese morphological and semantic relations. In: Proceedings of the 56th annual meeting of the association for computational linguistics, vol 2 (Short Papers). Association for Computational Linguistics, Melbourne, Australia, pp 138–143. https://doi.org/10.18653/v1/P18-2023. https://aclanthology.org/P18-2023
55. Yamada I, Asai A, Shindo H, Takeda H, Matsumoto Y (2020) LUKE: Deep contextualized entity representations with entity-aware self-attention. In: Proceedings of the 2020 conference on empirical methods in natural language processing (EMNLP). Association for Computational Linguistics, Online, pp 6442–6454. https://doi.org/10.18653/v1/2020.emnlp-main.523. https://aclanthology.org/2020.emnlp-main.523
56. Li X, Feng J, Meng Y, Han Q, Wu F, Li J (2020) A unified MRC framework for named entity recognition. In: Proceedings of the 58th annual meeting of the association for computational linguistics. Association for Computational Linguistics, Online, pp 5849–5859. https://doi.org/10.18653/v1/2020.acl-main.519. https://aclanthology.org/2020.acl-main.519
57. Yuan H, Yuan Z, Gan R, Zhang J, Xie Y, Yu S (2022) BioBART: pretraining and evaluation of a biomedical generative language model. arXiv . https://doi.org/10.48550/ARXIV.2204.03905. https://arxiv.org/abs/2204.03905
58. Wang Y, Shindo H, Matsumoto Y, Watanabe T (2021) Nested named entity recognition via explicitly excluding the influence of the best path. In: Proceedings of the 59th annual meeting of the association for computational linguistics and the 11th international joint conference on natural language processing, vol 1 (Long Papers). Association for Computational Linguistics, Online, pp 3547–3557. https://doi.org/10.18653/v1/2021.acl-long.275. https://aclanthology.org/2021.acl-long.275
59. Wang S, Li X, Meng Y, Zhang T, Ouyang R, Li J, Wang G (2022) kNN-NER: named entity recognition with nearest neighbor search . https://doi.org/10.48550/arXiv.2203.17103
60. Hu D, Wei L (2020) Slk-ner: Exploiting second-order lexicon knowledge for Chinese ner. In: Software engineering and knowledge engineering. https://doi.org/10.48550/arXiv.2007.08416

61. Chen C, Kong F (2021) Enhancing entity boundary detection for better Chinese named entity recognition. In: Proceedings of the 59th annual meeting of the association for computational linguistics and the 11th international joint conference on natural language processing, vol 2, (Short Papers). Association for Computational Linguistics, Online, pp 20–25. https://doi.org/10.18653/v1/2021.acl-short.4. https://aclanthology.org/2021.acl-short.4

62. Sun Z, Li X, Sun X, Meng Y, Ao X, He Q, Wu F, Li J (2021) ChineseBERT: Chinese pretraining enhanced by glyph and Pinyin information. In: Proceedings of the 59th annual meeting of the association for computational linguistics and the 11th international joint conference on natural language processing, vol 1, (Long Papers). Association for Computational Linguistics, Online, pp 2065–2075. https://doi.org/10.18653/v1/2021.acl-long.161. https://aclanthology.org/2021.acl-long.161

63. Nie Y, Tian Y, Song Y, Ao X, Wan X (2020) Improving named entity recognition with attentive ensemble of syntactic information. In: Empirical methods in natural language processing. https://doi.org/10.48550/arXiv.2010.15466

64. Teed Z, Deng J (2022) Raft: recurrent all-pairs field transforms for optical flow. https://doi.org/10.48550/arXiv.2003.12039

Publisher’s Note Springer Nature remains neutral with regard to jurisdictional claims in published maps and institutional affiliations.

Springer Nature or its licensor (e.g. a society or other partner) holds exclusive rights to this article under a publishing agreement with the author(s) or other rightsholder(s); author self-archiving of the accepted manuscript version of this article is solely governed by the terms of such publishing agreement and applicable law.