Span Graph Transformer for Document-Level Named Entity Recognition

Hongli Mao[1], Xian-Ling Mao[1]*, Hanlin Tang[1], Yu-Ming Shang[2], Heyan Huang[1]

[1]School of Computer Science & Technology, Beijing Institute of Technology, Beijing, China
[2] Beijing University of Posts and Telecommunications, Beijing, China
maohongli.bit@gmail.com, {maoxl, hltang, hhy63}@bit.edu.cn, shangym@bupt.edu.cn

Abstract

Named Entity Recognition (NER), which aims to identify the span and category of entities within text, is a fundamental task in natural language processing. Recent NER approaches have featured pre-trained transformer-based models (e.g., BERT) as a crucial encoding component to achieve state-of-the-art performance. However, due to the length limit for input text, these models typically consider text at the sentence-level and cannot capture the long-range contextual dependency within a document. To address this issue, we propose a novel **S**pan **G**raph **T**ransformer (**SGT**) method for document-level NER, which constructs long-range contextual dependencies at both the token and span levels. Specifically, we first retrieve relevant contextual sentences in the document for each target sentence, and jointly encode them by BERT to capture token-level dependencies. Then, our proposed model extracts candidate spans from each sentence and integrates these spans into a document-level span graph, where nested spans within sentences and identical spans across sentences are connected. By leveraging the power of Graph Transformer and well-designed position encoding, our span graph can fully exploit span-level dependencies within the document. Extensive experiments on both resource-rich nested and flat NER datasets, as well as low-resource distantly supervised NER datasets, demonstrate that proposed SGT model achieves better performance than previous state-of-the-art models.

Introduction

Named Entity Recognition (NER) is a basic task for Natural Language Processing (NLP), which aims to locate named entities within text and classify them into predefined entity types. As a subtask of information extraction, NER serves as an essential component in numerous NLP downstream tasks, including entity linking (Ganea and Hofmann 2017; Le and Titov 2018), relation extraction (Zhong and Chen 2021; Rathore et al. 2022) and knowledge graph construction (Sarhan and Spruit 2021).

The initial approaches for NER utilized feature-based models (Passos, Kumar, and McCallum 2014; Luo et al. 2015) while recent works have focused on deep learning methods (Chiu and Nichols 2016; Lample et al. 2016). Similar to other NLP tasks, the current approaches for NER employ pre-trained transformer architectures, like BERT (Devlin et al. 2019), as a key encoding component to achieve state-of-the-art (SOTA) results. Among these methods, classic sequence labeling methods (Souza, Nogueira, and Lotufo 2019; Li et al. 2020) combine BERT with CRF (Ma and Hovy 2016) to assign the BIO tag for each token. In a different approach, sequence-to-sequence frameworks (Yan et al. 2021; Fei et al. 2021) formulate NER as the span generation task, leveraging the power of BART (Lewis et al. 2020) to generate the entity sequence. Span-based methods (Yu et al. 2020; Zhu and Li 2022; Yan et al. 2022; Li et al. 2022), on the other hand, utilize BERT to generate contextualized span representation, achieving best performance for both flat and nested NER.

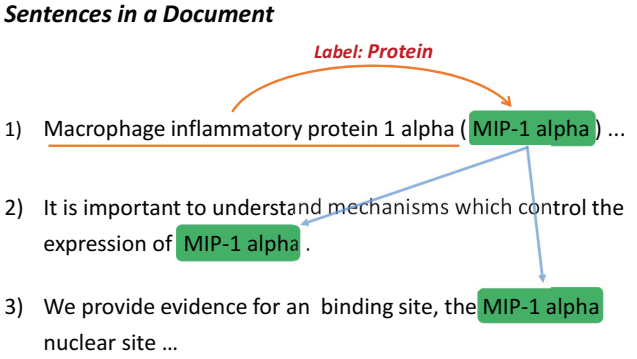

Figure 1: An motivating example from Genia dataset. The definition of "MIP-1 alpha" in sentence 1 implies it is a protein entity. By sharing this information across identical mentions via document-level dependencies, we can correctly predict the named entities of "MIP-1 alpha" in sentences 2 and 3 whereas sentence-level methods cannot.

However, as BERT can only encode input texts with up to 512 sub-tokens (due to the quadratic self-attention complexity), these BERT-based models typically process the long document sequentially at the sentence-level, failing to capture the long-range contextual dependency within a document. This limitation becomes especially apparent in low-resource settings, where insufficient training data hinders models from learning robust representation. Relying solely on sentence-level features may not provide enough informa-

*Corresponding author.

tion for precise entity identification. Although some works (Yamada et al. 2020; Amalvy et al. 2023) have attempted to incorporate the context of surrounding sentences, they still cannot explicitly model long-range document-level dependencies among spans, remaining constrained to sentence-level dependencies.

Inspired by the observation that identical spans within a document often refer to the same entity, we propose leveraging the label consistency between same spans to capture document-level contextual dependencies. For example, as shown in Figure 1, the first occurrence of the span "MIP-1 alpha" can provide useful information for disambiguating the second and third occurrences. The key idea here is to use identical spans to guide document-level representation learning, rather than isolating each occurrence. To this end, we can allow identical spans to share contexts, thereby enriching token-level representations. Moreover, by enabling deeper interaction among these spans' representations, we can explicitly model span-level dependencies to further improve robustness and accuracy of NER models.

Based on above considerations, we propose a novel **S**pan **G**raph **T**ransformer (**SGT**) method for document-level NER, which leverages label consistency to incorporate both token and span-level dependencies. Specifically, SGT is composed of two modules: the token-level dependency extractor and the span-level dependency extractor. First, for each target sentence, the token-level module selects several top-relevant sentences from a document based on span overlap and inter-sentence distance, allowing identical spans to share context across sentences. The target sentence and corresponding context are then jointly fed into BERT to capture token-level dependencies. Second, the span-level module applies the biaffine method to generate span representations and extracts candidate spans from each sentence. Following this, these spans are integrated into a document-level graph, which creates connections between nested spans within sentences and identical spans across sentences. To capture span-level dependencies via graph, we introduce a Graph Transformer equipped with well-designed position encoding to effectively encode the graph structure and model interactions between spans. Extensive experiments on resource-rich nested (ACE 2004, ACE 2005, Genia) and flat (CoNLL 2003, OntoNotes 5) NER datasets, as well as low-resource distantly supervised NER dataset (BC5CDR) validate the effectiveness of our proposed model in achieving new SOTA results.

In summary, our main contributions are as follows:

- Unlike previous methods that model NER at the sentence level, we leverage label consistency to capture both token and span-level contextual dependencies for document-level NER.
- We formulate the document as a span graph, and introduce the Graph Transformer with an ingenious position encoding to effectively incorporate graph structure information and model interactions between spans.
- We conduct experiments in a variety of settings, and we obtain a 0.5%-1.8% absolute improvement on both standard nested and flat NER datasets. In the low-resource distantly supervised NER, our model still exceeds other SOTA methods by a margin of 2.3%.

Related Work

Named Entity Recognition

Sentence-Level NER Recently, deep learning methods have demonstrated effectiveness on NER tasks (Chiu and Nichols 2016; Lample et al. 2016). Transformer-based language models like BERT serve as key components of current SOTA deep learning models for NER. For flat NER, traditional methods apply BERT with CRF for sequence labeling to achieve best accuracy (Souza, Nogueira, and Lotufo 2019; Li et al. 2020). For nested NER, sequence-to-sequence frameworks utilize BART to generate both entity mentions and their types (Yan et al. 2021; Fei et al. 2021). While span-based methods typically enumerate all candidate spans and employ BERT to generate span representations for classification (Yu et al. 2020; Zhu and Li 2022; Yan et al. 2022). And the contrastive learning frameworks (Zhang et al. 2023) leverage BERT to map spans and entity types into the same representation space. However, these methods model NER at sentence-level, cannot capture document-level features.

Document-Level NER There have been a few recent attempts to apply document-level contextualized information for NER. Early methods leverage LSTMs to encode long documents but cannot benefit from pre-trained contextual embeddings (Hu et al. 2020; Gui et al. 2021). Due to input length limits, most BERT-based models incorporate surrounding sentences to complement local feature (Yamada et al. 2020; Amalvy et al. 2023), but they remain constrained to sentence-level and cannot extract dependencies within spans across the full document. Although GPT-based models can encode long contexts, their generative architecture struggles to extract entities and underperform BERT-based models (Qin et al. 2023). In our setting, to capture document-level dependencies, we allow identical spans to share contexts across sentences and formulate the document as a span graph to update span representations.

Distantly Supervised NER In the low-resource distantly supervised setting, annotated training datasets are automatically generated by string matching with the dictionary. Due to limited coverage of the dictionary, many latent entities are not extracted, hindering models from learning robust representation. To address this limitation, Mao et al.(Mao et al. 2020) leverage self-learning to iteratively mine latent entities. Bond (Liang et al. 2020) also adopt a iterative teacher-student self-training framework to refresh training datasets. Zhou et al. (Zhou, Li, and Li 2022) leverage a two-step Positive and Unlabeled (PU) learning, adding additional entity confidence to each token. Without additional iterative training, our method directly utilizes the document-level features to resolve the ambiguity of the entity and improve the robustness of representation learning.

Graph Transformer

For Graph Transformer, it has been used in graph representation tasks to help alleviate the problems of over-

smoothing and over-squashing in Graph Neural Networks (GNNs) (Alon and Yahav 2021). Dwivedi et al. (Dwivedi and Bresson 2020) introduce a Transformer based on GNNs, restricting attention to node neighborhoods. Based on this, SAN (Kreuzer et al. 2021) propose a full Laplacian spectrum to encode node positions. Further, Graphormer (Ying et al. 2021) propose using pair-wise graph distances to define positional encoding, then incorporate node and edge features into the model, achieving outstanding success on this benchmarks. To the best of our knowledge, we are the first to apply a Graph Transformer for NER. We extract an undirected span graph from the document, where node features are produced from BERT and a biaffine model. In addition, we design a novel relative position encoding to encode the span graph structure.

The Proposed Method

The architecture of our proposed SGT model is shown in Figure 2. It consists of two main parts. First, the token-level dependency extractor retrieves the most relevant context sentences for each target sentence, and encodes them jointly by BERT to get contextual token embeddings. Next, span-level dependency extractor feeds these token embeddings into a biaffine encoder to generate span representations for extracted candidate spans. Important dependencies between these spans are integrated into a document-level graph and modeled using a Graph Transformer module. In the following, we will first provide a task definition, then present the model components in detail.

Task Formulation

The input to SGT is a document $D = [S_1, S_2, \ldots, S_n]$, $S_i = [w_{i1}, w_{i2}, \ldots, w_{i,o_i}]$ $(1 \leq i \leq n)$ where n denotes the total number of sentences in the document and o_i denotes the number tokens in the sentence S_i. The length of the document is defined as the sum of length of each sentence: $d_n = \sum_i^n o_i$, where d_n is always larger than 512. The goal of SGT is to extract all entities $E = \{(l_i, r_i, t_i)\}_{i=0}^{e_n}$ based on the document context beyond individual sentences, where e_n is the number of entities and l_i, r_i, t_i represent the left and right boundary indices and type of the i-th entity.

Token-Level Dependency Extractor

As discussed in the introduction, identical entity mentions within a document frequently refer to the same entity. To enable contextual sharing between these repeated entity references, we propose selecting the most relevant sentences from the document as context for each target sentence.

Context Selection We assess contextual sentence relevance based on two criteria: span overlap and inter-sentence distance. Sentences with greater span overlap are more likely to contain related entities, thus providing useful contextual information. Additionally, under similar overlap conditions, sentences in closer proximity tend to be more topically related to the target. To generate sentence spans, we leverage the constituency parsing tree[1] to extract all tree nodes as spans set $\mathrm{SpanEx}(S_i)$ for sentence S_i. The distance metric is then converted to a weight score by utilizing a Gaussian function. Formally, the relevant score t_{ij} for context sentence S_j and target S_i is computed as:

$$\begin{aligned} t_{ij}^s &= \frac{|\mathrm{SpanEx}(S_i) \cap \mathrm{SpanEx}(S_j)|}{|\mathrm{SpanEx}(S_j)|} \\ t_{ij}^d &= \frac{1}{\sigma\sqrt{2\pi}} e^{-\frac{1}{2}\left(\frac{|i-j|}{\sigma}\right)^2} \\ t_{ij} &= \alpha \odot t_{ij}^s + (1-\alpha) \odot t_{ij}^d \end{aligned} \tag{1}$$

where α is the weight of span overlap score t^s, $\odot$ denotes an operator for element-wise production. In the Gaussian function, we set the mean $\mu = 0$, leverage the standard deviation σ to control the rate of decay for the score t^d with distance.

BERT Encoder Based on the relevant score t, we can retrieve a set C_i containing the top-ranking context sentences for the target sentence S_i in document D, i.e., $C_i \subset S_{context} = \{S_j \in D \mid j \neq i\}$. The target sentence S_i is then augmented with the sentences in C_i to form an expanded context S_i' (i.e., $S_i' = S_i \cup C_i$). In our setting, k_n(the length of $S_i^{'}$) is constrained to fall within BERT's input length limit. The sentences in S_i' are sorted by their original position order and concatenated as:

$$S_i^{'} = \left[w_{cls}, C_i^L, w_{sep}, S_i, w_{sep}, C_i^R\right] \tag{2}$$

where $C_i^L = \{S_j \in C_i \mid j < i\}$, $C_i^R = \{S_j \in C_i \mid j > i\}$ represent the context sentences appearing left and right S_i respectively. The tokens w_{cls}, w_{sep} correspond to the start and separator tokens in BERT.

Then we can encode the expanded context $S_i^{'}$ entirely into BERT to get token-level representation $\mathbf{H}_i = [h_{i1}, h_{i2}, \ldots, h_{i,oi}]$ for sentence S_i. Leveraging the shared context and BERT's self-attention mechanism, we can directly incorporate token-level dependencies into $\mathbf{H}_i$.

Span-Level Dependency Extractor

Relying solely on BERT for token-level interactions cannot explicitly capture deeper dependencies that exist at the span-level. To address this, we construct connections between entity candidate spans in a document graph and apply a Graph Transformer to incorporate span interactions.

Span Graph Node Initialization After getting the contextualized embedding of tokens from BERT, we follow (Yu et al. 2020) and leverage a biaffine encoder to generate span representation. Specifically, for sentence S_k, the token embedding $\mathbf{H}_k$ is fed into two Feed Forward Networks (FFN) to respectively obtain the start of span representations $\mathbf{H}_k^s \in \mathbb{R}^{o_k \times d_h}$ and end of span representations $\mathbf{H}_k^e \in \mathbb{R}^{o_k \times d_h}$, then the span representation matrix $\mathbf{R}_k$ can be computed as:

$$\begin{aligned} \mathbf{H}_k^s &= \mathrm{FFN_s}(\mathbf{H}_k) \\ \mathbf{H}_k^e &= \mathrm{FFN_e}(\mathbf{H}_k) \\ \mathbf{R}_k &= (\mathbf{H}_k^s)^T \mathbf{U} \mathbf{H}_k^e + \mathbf{W}(\mathbf{H}_k^s \oplus \mathbf{H}_k^e) + \mathbf{b} \end{aligned} \tag{3}$$

[1] we utilize one of the SOTA parsers (Kitaev and Cao 2019) to generate the constituency parsing tree for each sentence

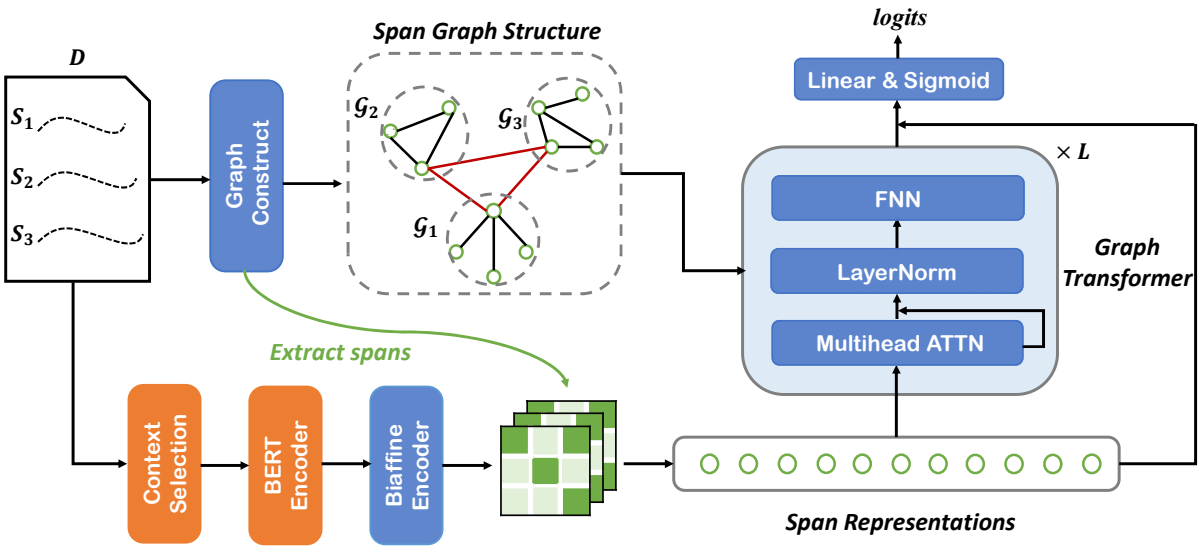

Figure 2: The overall architecture of our proposed model.

where $\mathbf{U} \in \mathbb{R}^{d_h \times d_r \times d_h}$, $\mathbf{W} \in \mathbb{R}^{d_r \times 2d_h}$ and $\mathbf{b} \in \mathbb{R}^{d_r}$ are learnable parameters, d_h and d_r are the hidden size, $\oplus$ denotes an operator for element-wise concatenation, and $\mathbf{R}_k \in \mathbb{R}^{o_k \times o_k \times d_r}$. Each cell (i, j) in the matrix $\mathbf{R}_k$ represents the feature representation of span starting at the $i-$th token and ending at the $j-$th token in sentence S_k.

The biaffine model efficiently generates spans representations matrix in parallel, we then leverage the constituency parsing tree to extract tree nodes as candidate spans. This can filter out numerous spans lacking valid syntax while retaining entities, thereby better capturing span dependencies. Specifically, we construct the parsing tree using a SOTA parser (Kitaev and Cao 2019). Since some subtrees directly branching into leaves may miss short entities, we generate additional nodes under these subtrees by enumeration.

Based on the parsing tree, we extract candidate spans from each sentence and concatenate them to form the set C comprising all candidate spans in the document, where $C = \{c_1, c_2, \ldots, c_m\}$ and m is the total number of spans. For a given span c_i, we use $\text{order}[i]$ to denote the index of its corresponding sentence, and $\text{start}[i]$, $\text{end}[i]$ to denote its start and end token indices within that sentence. Then, its span representation can be denoted as $\mathbf{r}_i = \mathbf{R}_{\text{order}[i]}(\text{start}[i], \text{end}[i])$.

Span Graph Edge Construction In NLP transformers, a sentence, usually with less than tens or hundreds of tokens, is modeled as a fully connected graph with self-attention automatically capturing key word dependencies. However, the number of candidate spans in our documents can reach several thousand. Using fully connected graph to capture potential dependencies would be difficult. Therefore, we construct a sparse graph with only two types of undirected edges that connect highly related spans:

- **Nested Mention Edges**: Connect nested mentions within sentences to incorporate local context.
- **Identical Mention Edges**: Link identical mentions across sentences to enable document-level interactions.

In this manner, we inject prior knowledge into the graph structure, augmenting the model's capacity for capturing important dependencies.

Graph Transformer Architecture To capture span-level dependency in the graph, we use Graph Transformer to aggregate neighbor messages to update the node representation. Our Graph Transformer is built upon the original implementation of classic Transformer encoder(Vaswani et al. 2017). We now proceed to define node update equations for a layer ℓ.

$$\mathbf{e}_{ij} = \frac{\left(\mathbf{r}_i^{\ell}\mathbf{W}^{Q}\right)\left(\mathbf{r}_j^{\ell}\mathbf{W}^{K}\right)^{\mathrm{T}}}{\sqrt{d_z}} \tag{4}$$

where e_{ij} denotes attention score from node j to node i, To normalize them, there is:

$$\alpha_{ij} = \text{softmax}_j\left(e_{ij}\right) = \frac{\exp\left(e_{ij}\right)}{\sum_{k \in \mathcal{N}_i} \exp\left(e_{ik}\right)} \tag{5}$$

Then $\mathbf{r}_i^{\ell+1}$, the representation of node i in layer $\ell + 1$, can be computed as :

$$\mathbf{z}_i^{\text{head}_m} = \sum_{j \in \mathcal{N}_i} \alpha_{ij} \mathbf{r}_j \mathbf{W}^{V} \tag{6}$$

$$\hat{\mathbf{r}}_i^{\ell+1} = \text{Cat}\left[\mathbf{z}_i^{\text{head}_1}, \ldots, \mathbf{z}_i^{\text{head}_k}\right] \tag{7}$$

$$\mathbf{r}_i^{\ell+1} = \text{FFN}(\text{LayerNorm}(\mathbf{r}_i^{\ell} + \hat{\mathbf{r}}_i^{\ell+1})) \tag{8}$$

where $\mathbf{W}^{Q}$, $\mathbf{W}^{K}$, $\mathbf{W}^{V}$ are the parameters of the attention query, key and value of head_m, d_z is dimension of each head, $\mathcal{N}_i$ is the set of target node i's neighbours. $\mathbf{z}_i^{\text{head}_m}$ is the output of head_m in the multi-head attention, and all these outputs are then concatenated into the vector $\hat{\mathbf{r}}_i^{\ell+1}$. Finally, $\mathbf{r}_i^{l+1}$ is obtained by passing $\hat{\mathbf{r}}_i^{\ell+1}$ through a feedforward layer with residual connections and layer normalization.

Relative Position Encoding The document-level graph contains candidate spans from each sentence. To encode the interactions among spans, we propose the relative position encoding of spans. Concretely, we use three relative positions to indicate the relation between span c_i and span c_j:

$$\begin{aligned} d_{ij}^{o} &= \text{order}[i] == \text{order}[j] \\ d_{ij}^{s} &= \text{start}[i] - \text{start}[j] \\ d_{ij}^{e} &= \text{end}[i] - \text{end}[j] \end{aligned} \tag{9}$$

We employ d_{ij}^{o} to denote whether spans c_i and c_j originate from the same sentence. For spans nested within a sentence, we utilize relative distances d_{ij}^{s} and d_{ij}^{e} to depict the degree of nesting between them. In particular, for identical spans across sentences, we manually assign $d_{ij}^{s} = d_{ij}^{e} = 0$ to signify their complete overlap.

Inspired by T5 model (Raffel et al. 2020), we convert three relative positional features into learnable low-dimensional embeddings, and fuse them via nonlinear transformation to obtain a scalar representation of the final relative position encoding:

$$b_{ij} = \text{ReLU}\left(\mathbf{W}_b\left(\mathbf{B}^1_{d^o_{ij}} \oplus \mathbf{B}^2_{d^s_{ij}} \oplus \mathbf{B}^3_{d^e_{ij}}\right)\right) \tag{10}$$

where $\mathbf{B}$ are learnable embeddings shared across all layers. The scalar b_{ij} then serves as a bias term in the self-attention module to effectively encode the graph structure:

$$\mathbf{e}^{*}_{ij} = \frac{\left(\mathbf{r}^{\ell}_i \mathbf{W}^{Q}\right)\left(\mathbf{r}^{\ell}_j \mathbf{W}^{K}\right)^{\mathrm{T}}}{\sqrt{d_z}} + b_{ij} \tag{11}$$

We replace $\mathbf{e}_{ij}$ with $\mathbf{e}^{*}_{ij}$ in Eq.(4). The remaining computations follow the standard Graph Transformer to produce the final span representations $\mathbf{r}^{L}_i$ after L layers.

Training and Inference

In the training process, non-graph nodes filtered out by the parsing tree in the span matrix $\mathbf{R}$ are also included to assist the biaffine encoder in generating better representations. For each span(k, i, j) in the sentence k, the prediction logits $\mathbf{P}^{k}_{ij} \in \mathbb{R}^{c}$ (c is the number of entity types) can be calculated as follow:

$$\mathbf{S}^{k}_{ij} = \begin{cases} \mathbf{R}_k(i,j) + \mathbf{R}^{L}_k(i,j), & \text{Graph node} \\ \mathbf{R}_k(i,j), & \text{Non-graph node} \end{cases} \tag{12}$$

$$\mathbf{P}^{k}_{ij} = \text{Sigmoid}(\mathbf{W}_p \mathbf{S}^{k}_{ij} + \mathbf{b}_p) \tag{13}$$

where $\mathbf{R}_k(i,j)$ is generated by biaffine encoder, $\mathbf{R}^{L}_k(i,j)$ is generated by L layers Graph Transformer, $\mathbf{W}_p \in \mathbb{R}^{c \times d_r}$, $\mathbf{b}_p \in \mathbb{R}^{c}$. And then, we use the binary cross entropy to calculate the loss as:

$$\mathcal{L}_{BCE} = -\sum_{0 \le k < n} \sum_{0 \le i \le j < o_k} y^{k}_{ij} \log\left(\mathbf{P}^{k}_{ij}\right) \tag{14}$$

In the inference, we first prune out non-entity spans by setting a threshold θ. We then greedily select the highest probability entity proposal, ignoring conflicting proposals whose boundaries clash with chosen entities, until all proposals are processed.

Experiments

Experimental Setting

Datasets In the resource-rich setting, we perform experiments on three nested NER datasets: ACE 2004[2], ACE 2005[3] and Genia (Kim et al. 2003); and two flat NER datasets CoNLL 2003 (Sang and De Meulder 2003) and OntoNotes 5[4]. For ACE 2004 and ACE 2005, we follow the same settings of Lu and Roth (2015) and Muis and Lu (2017) to split the data into train, dev and test sets by 8:1:1. For Genia, we use the same document split as suggested as Lu and Roth (2015) and Yu et al. (2020). For flat NER CoNLL2003 and OntoNotes 5, preprocessing and splits follow Yu et al. (2020). For the low-resource distantly supervised setting, we use BC5CDR dataset with standard train, dev, and test splits. The distant labels are generated by exact string matching against a entity dictionary released by Shang et al. (2018).

Implementation Detail For a fair comparison, following previous SOTA works (Zhu and Li 2022; Yan et al. 2022), we use RoBERTa-base (Liu et al. 2019) on ACE 2005, ACE 2004, CoNLL2003 and OntoNotes 5, biobert-base (Lee et al. 2020) on Genia and BC5CDR. For SGT model, the weight of span overlap α is set as 0.9, the length of the expanded context k_n is 50. The number of Graph Transformer layer L is 4, filtering threshold θ is set as 0.5. All parameters are optimized using Adam with a peak learning rate of $2e-5$. Final reported results are averages over three runs with different random seeds.

Main Results

Supervised Nested and Flat NER The comparison between SGT and the baselines on the nested NER datasets is presented in Table 1. As is shown, our method achieves the best performance on nested NER. Specifically, we obtain +0.59% and +0.95% F1 increases on ACE 2004 and 2005 respectively. Notably, we significantly surpass previous methods by +1.83% on Genia. We further evaluate our framework on two flat NER datasets, as shown in table 2. Again, our approach outperforms previous SOTA methods on flat NER, achieving an impressive F1 increase of +0.54% on CoNLL 2003 and +0.57% on OntoNotes 5. These improvements demonstrate the necessity of modeling NER from the document perspective. Although the GPT-NER based on GPT-3.5 can encode long context, their generative architecture struggles to perceive entity boundaries and fails to explicitly capture dependencies between entities, substantially underperforming our model. By incorporating token and span-level dependencies into representation of entities, SGT obtains superior performance on both nested and flat NER tasks.

Low-Resource Distantly Supervised NER Table 3 illustrates the performance of the proposed model as well as baselines on BC5CDR datasets. We observe that SGT achieves the best performance on both low and rich resource types. In particular, SGT makes a significant improvement

[2]https://catalog.ldc.upenn.edu/LDC2005T09
[3]https://catalog.ldc.upenn.edu/LDC2006T06
[4]https://catalog.ldc.upenn.edu/LDC2013T19

Method	ACE 2004			ACE 2005			Genia		
	Pre	Rec	F1	Pre	Rec	F1	Pre	Rec	F1
Biaffine (Yu et al. 2020)	87.30	86.00	86.70	85.20	85.60	85.40	81.80	79.30	80.50
Seq2Seq (Yan et al. 2021)	87.27	86.41	86.84	83.16	86.38	84.74	78.87	79.60	79.23
Locate&Label (Shen et al. 2021)	87.44	87.38	87.41	86.09	87.27	86.67	80.19	80.89	80.54
Parsing-Tree (Lou et al. 2022)	87.39	88.40	87.90	85.97	87.87	86.91	78.39	78.50	78.44
W^2NER (Li et al. 2022)	87.33	87.71	87.52	85.03	88.62	86.79	83.10	79.76	81.39
BS (Zhu and Li 2022)	88.43	87.53	87.98	86.25	88.07	87.15	-	-	-
Biaffine-CNN(Yan et al. 2022)	87.33	87.29	87.31	86.70	88.16	87.42	81.52	79.17	80.33
GPT-NER (Wang et al. 2023)	73.29	75.11	74.20	72.77	75.51	73.59	61.89	66.95	64.42
DiffusionNER(Shen et al. 2023)	88.11	88.66	<u>88.39</u>	86.15	87.72	86.93	82.10	80.97	<u>81.53</u>
BINDER (Zhang et al. 2023)†	87.20	87.60	87.40	87.90	88.40	<u>88.20</u>	83.40	78.30	80.80
SGT(Ours)	89.07	88.89	**88.98**	89.06	89.33	**89.15**	84.25	82.49	**83.36**

Table 1: Results on the three nested entity datasets, Bold and underline indicate the best and the second best F1 score respectively. † We reproduce BINDER using the same data split and same pre-trained model.

Method	CoNLL 2003			OntoNotes 5		
	Pre	Rec	F1	Pre	Rec	F1
Biaffine (Yu et al. 2020)	93.70	93.30	<u>93.50</u>	91.10	91.50	91.30
DocL-NER (Gui et al. 2021)	-	-	93.05	-	-	88.49
BS (Zhu and Li 2022)†	93.15	93.48	93.41	91.18	91.52	<u>91.34</u>
GPT-NER (Wang et al. 2023)	89.76	92.06	90.91	79.89	84.51	82.20
DiffusionNER(Shen et al. 2023)	92.99	92.56	92.78	90.31	91.02	90.66
BINDER (Zhang et al. 2023)	93.08	93.57	93.33	-	-	-
SGT(Ours)	93.90	94.18	**94.04**	91.59	92.31	**91.91**

Table 2: Results on the two flat entity datasets. † denotes our reproduction of BS under the same setting.

in low-resource distantly supervised setting, exceeding other SOTA methods by a margin of 2.31%. In low-resource settings, dictionary-based distant labeling methods suffer from low entity recall due to limited dictionary coverage. Training on such a limited coverage dataset can hinder models from learning robust representation. Compared to previous SOTA works operating at the sentence level, SGT leverages more document-level features to enhance potential entity confidence, thereby substantially improving recall by 6.57% while preserving high precision. This demonstrates that SGT can learn more robust representation and is less affected by this sparsely labeled data by complementing limited supervision with rich document-level dependencies.

Ablation Studies

In this section, we analyze the effects of different components in SGT. As shown in Table 4, the following observations can be found: (1) Removing either token or span-level dependencies leads to varying degrees of performance drops, indicating both are critical for capturing long-range document dependencies. The token module provides a contextual foundation via BERT. The span module then enables deeper integration upon these embeddings to capture document dependencies. (2) On the biomedical datasets Genia and BC5CDR, removing span-level dependencies causes

Method	BC5CDR		
	Pre	Rec	F1
Distantly Supervised			
Dict Matching	86.39	51.24	64.32
AutoNER (Shang et al. 2018)	82.63	77.52	79.99
Bond (Liang et al. 2020)	80.43	67.94	73.66
BINDER (Zhang et al. 2023)	87.60	76.30	<u>81.60</u>
SGT(Ours)	84.97	82.87	**83.91**
Fully Supervised			
Wang et al. (2021)	-	-	90.99
BINDER (Zhang et al. 2023)	92.60	91.20	<u>91.90</u>
SGT(Ours)	90.95	93.53	**92.22**

Table 3: Results on Distantly Supervised NER datasets.

performance drops of 2.87% and 1.44% respectively, compared to decreases of only 0.39% and 0.36% when eliminating the token module. It illustrates the importance of deeper integration of information between entity candidate spans. In these specific domains, entities tend to be longer and require more direct span interactions for disambiguation.

Model	GENIA	CoNLL03	BC5CDR
Default	**83.36**	**94.04**	**83.91**
w/o Token	82.97	93.52	83.55
w/o Span	80.49	93.64	82.47
w/o Token&Span	79.58	92.89	81.52

Table 4: Ablation Study. (1) w/o Token: remove token-level dependency, i.e., remove external contexts for token embedding. (2) w/o Span: remove span-level dependency, i.e., eliminate Graph Transformer and directly employ biaffine for span embedding generation. (3) w/o Token&Span: remove both token and span-level dependencies.

	SimCSE	SD	SO	SD+SO
Genia	82.98	83.09	83.21	**83.36**
CoNLL03	93.76	93.92	93.87	**94.04**

Table 5: A comparison of different context selection approaches by the F1 scores. (1) SD: Sentence Distance score. (2) SO: Span Overlap score.

Analysis

Analysis of Context Selection To demonstrate the effect of context sentence selection, we compare our approach with three other methods. As shown in Table 5, SimCSE (Gao, Yao, and Chen 2021) method which relies on sentence vector similarity to retrieve contexts yields the lowest performance. For span-based NER, such coarse-grained relevance ranking doesn't seem appropriate. The SO method selects high span overlap sentences from a fine-grained perspective, while closer sentences exhibit higher topical relatedness. Compared to using either signal alone, our joint method (SD+SO) achieves the best context selection by combining fine-grained semantic similarity with locality awareness.

	Genia	BC5CDR
Default	**83.36**	**83.91**
Full connected graph	81.96	83.10
Same connected graph	82.08	82.72
Without graph	80.49	82.47

Table 6: A comparison of different graph connections (1) Full connected graph: connect all candidate spans in document. (2) Same connected graph: connect same spans in document. (3) Without graph: remove Graph Transformer.

Analysis of Graph Connection We further investigate the effect of different graph connections. The results are shown in Table 6. All graph-based models outperform the setting without graph modeling, validating the importance of span interactions. Although a fully connected graph enables global attention across all spans, the quadratic expansion in edges as span number increases in the document makes it difficult to automatically focus on the most relevant dependencies. In our approach, we prioritize attention to nested spans within sentences and identical spans across sentences, establishing a sparsely-connected graph. This selective connectivity effectively facilitates information integration between relevant spans, achieving better results. In contrast, the same connected graph that only links identical mentions creates disjoint subgraphs, missing potentially relevant indirect dependencies between spans, and is less effective than our method.

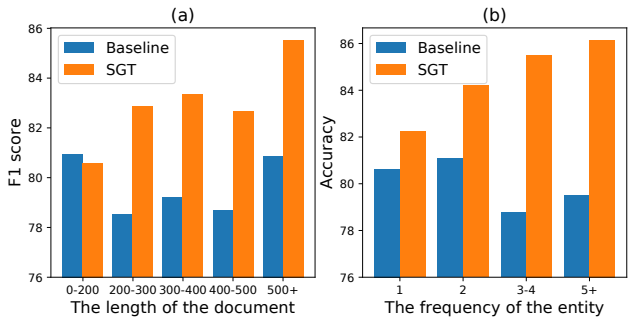

Figure 3: The performance on Genia dataset, broken down by document length (a) and entity frequency (b). Baseline: refer to remove token and span level dependency modules of SGT, directly use the span-based biaffine model.

Analysis of Long-Range Dependency As previously highlighted, our proposed SGT aims to leverage label consistency to capture long-range dependency within the document. To validate this point, we compare the performance of SGT against the baseline on different document lengths and entity frequencies. As shown in Figure 3(a), we can observe that the sentence-level baseline remains relatively stable as document length increases, while the performance gain of SGT is more significant. This demonstrates SGT can effectively extract document-level features to model long-range dependencies. Figure 3(b) further validates our hypothesis. For entities appearing multiple times (≥ 2) in a document, we leverage the token and span-level dependency modules to integrate contextual information between these repeated mentions, acquiring more representational features. This increases the confidence of entities, substantially boosting SGT's performance over the baseline.

Conclusion

In this paper, we propose a novel method SGT to incorporate both token and span-level dependencies for document-level NER. Our model retrieve important context sentences for the target sentence to incorporate token-level dependencies. To capture span-level dependencies, we formulate the document as a span graph and design a Graph Transformer with an ingenious position encoding to model interactions between spans. We conduct extensive experiments in various settings including nested and flat NER as well as low-resource distantly supervised NER, demonstrating that our approach significantly outperforms previous state-of-the-art methods across all datasets.

Acknowledgments

The work is supported by National Key R&D Plan (No. 2020AAA0106600), National Natural Science Foundation of China (No.62172039, U21B2009 and 62276110), and MIIT Program (CEIEC-2022-ZM02-0247).

References

Alon, U.; and Yahav, E. 2021. On the Bottleneck of Graph Neural Networks and its Practical Implications. In *International Conference on Learning Representations*.

Amalvy, A.; Labatut, V.; Dufour, R.; and Dufour, R. 2023. The Role of Global and Local Context in Named Entity Recognition. In *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics*, 714–722. Toronto, Canada: Association for Computational Linguistics.

Chiu, J. P.; and Nichols, E. 2016. Named entity recognition with bidirectional LSTM-CNNs. *Transactions of the association for computational linguistics*, 4: 357–370.

Devlin, J.; Chang, M.-W.; Lee, K.; and Toutanova, K. 2019. Bert: Pre-training of deep bidirectional transformers for language understanding. *arXiv preprint arXiv:1810.04805*.

Dwivedi, V. P.; and Bresson, X. 2020. A generalization of transformer networks to graphs. *arXiv preprint arXiv:2012.09699*.

Fei, H.; Ji, D.; Li, B.; Liu, Y.; Ren, Y.; and Li, F. 2021. Rethinking boundaries: End-to-end recognition of discontinuous mentions with pointer networks. In *Proceedings of the AAAI conference on artificial intelligence*, volume 35, 12785–12793.

Ganea, O.-E.; and Hofmann, T. 2017. Deep Joint Entity Disambiguation with Local Neural Attention. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*, 2619–2629. Copenhagen, Denmark: Association for Computational Linguistics.

Gao, T.; Yao, X.; and Chen, D. 2021. SimCSE: Simple Contrastive Learning of Sentence Embeddings. In *2021 Conference on Empirical Methods in Natural Language Processing, EMNLP 2021*, 6894–6910. Association for Computational Linguistics (ACL).

Gui, T.; Ye, J.; Zhang, Q.; Zhou, Y.; Gong, Y.; and Huang, X. 2021. Leveraging document-level label consistency for named entity recognition. In *Proceedings of the Twenty-Ninth International Conference on International Joint Conferences on Artificial Intelligence*, 3976–3982.

Hu, A.; Dou, Z.; Nie, J.-Y.; and Wen, J.-R. 2020. Leveraging multi-token entities in document-level named entity recognition. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 34, 7961–7968.

Kim, J.-D.; Ohta, T.; Tateisi, Y.; and Tsujii, J. 2003. GENIA corpus—a semantically annotated corpus for bio-textmining. *Bioinformatics*, 19(suppl_1): i180–i182.

Kitaev, N.; and Cao. 2019. Multilingual Constituency Parsing with Self-Attention and Pre-Training. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, 3499–3505.

Kreuzer, D.; Beaini, D.; Hamilton, W.; Létourneau, V.; and Tossou, P. 2021. Rethinking graph transformers with spectral attention. *Advances in Neural Information Processing Systems*, 34: 21618–21629.

Lample, G.; Ballesteros, M.; Subramanian, S.; Kawakami, K.; and Dyer, C. 2016. Neural Architectures for Named Entity Recognition. In *Proceedings of the 2016 Conference of the North American Chapter of the Association for Computational Linguistics*, 260–270.

Le, P.; and Titov, I. 2018. Improving Entity Linking by Modeling Latent Relations between Mentions. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, 1595–1604. Melbourne, Australia: Association for Computational Linguistics.

Lee, J.; Yoon, W.; Kim, S.; Kim, D.; Kim, S.; So, C. H.; and Kang, J. 2020. BioBERT: a pre-trained biomedical language representation model for biomedical text mining. *Bioinformatics*, 36(4): 1234–1240.

Lewis, M.; Liu, Y.; Goyal, N.; Ghazvininejad, M.; Mohamed, A.; Levy, O.; Stoyanov, V.; and Zettlemoyer, L. 2020. Bart: Denoising sequence-to-sequence pre-training for natural language generation, translation, and comprehension. *arXiv preprint arXiv:1910.13461*.

Li, J.; Fei, H.; Liu, J.; Wu, S.; Zhang, M.; Teng, C.; Ji, D.; and Li, F. 2022. Unified named entity recognition as word-word relation classification. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 36, 10965–10973.

Li, X.; Yan, H.; Qiu, X.; and Huang, X. 2020. FLAT: Chinese NER using flat-lattice transformer. *arXiv preprint arXiv:2004.11795*.

Liang, C.; Yu, Y.; Jiang, H.; Er, S.; Wang, R.; Zhao, T.; and Zhang, C. 2020. Bond: Bert-assisted open-domain named entity recognition with distant supervision. In *Proceedings of the 26th ACM SIGKDD international conference on knowledge discovery & data mining*, 1054–1064.

Liu, Y.; Ott, M.; Goyal, N.; Du, J.; Joshi, M.; Chen, D.; Levy, O.; Lewis, M.; Zettlemoyer, L.; and Stoyanov, V. 2019. Roberta: A robustly optimized bert pretraining approach. *arXiv preprint arXiv:1907.11692*.

Lou, C.; Yang, S.; Tu, K.; and Tu, K. 2022. Nested Named Entity Recognition as Latent Lexicalized Constituency Parsing. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics*, 6183–6198. Association for Computational Linguistics.

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

Luo, G.; Huang, X.; Lin, C.-Y.; and Nie, Z. 2015. Joint entity recognition and disambiguation. In *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing*, 879–888.

Ma, X.; and Hovy, E. 2016. End-to-end Sequence Labeling via Bi-directional LSTM-CNNs-CRF. In *Proceedings of the*

54th Annual Meeting of the Association for Computational Linguistics, 1064–1074.

Mao, H.; Tang, H.; Zhang, W.; Huang, H.; and Mao, X.-L. 2020. A Span-Based Distantly Supervised NER with Self-learning. In *Natural Language Processing and Chinese Computing: 9th CCF International Conference, NLPCC 2020, Zhengzhou, China, October 14–18, 2020, Proceedings, Part I 9*, 192–203. Springer.

Muis, A. O.; and Lu, W. 2017. Labeling Gaps Between Words: Recognizing Overlapping Mentions with Mention Separators. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*, 2608–2618.

Passos, A.; Kumar, V.; and McCallum, A. 2014. Lexicon Infused Phrase Embeddings for Named Entity Resolution. In *Proceedings of the Eighteenth Conference on Computational Natural Language Learning*, 78–86.

Qin, C.; Zhang, A.; Zhang, Z.; Chen, J.; Yasunaga, M.; and Yang, D. 2023. Is ChatGPT a general-purpose natural language processing task solver? *arXiv preprint arXiv:2302.06476*.

Raffel, C.; Shazeer, N.; Roberts, A.; Lee, K.; Narang, S.; Matena, M.; Zhou, Y.; Li, W.; and Liu, P. J. 2020. Exploring the limits of transfer learning with a unified text-to-text transformer. *The Journal of Machine Learning Research*, 21(1): 5485–5551.

Rathore, V.; Badola, K.; Singla, P.; et al. 2022. PARE: A Simple and Strong Baseline for Monolingual and Multilingual Distantly Supervised Relation Extraction. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, 340–354.

Sang, E. F.; and De Meulder, F. 2003. Introduction to the CoNLL-2003 shared task: Language-independent named entity recognition. *arXiv preprint cs/0306050*.

Sarhan, I.; and Spruit, M. 2021. Open-CyKG: An Open Cyber Threat Intelligence Knowledge Graph. *Knowledge-Based Systems*, 233: 107524.

Shang, J.; Liu, L.; Gu, X.; Ren, X.; Ren, T.; and Han, J. 2018. Learning Named Entity Tagger using Domain-Specific Dictionary. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, 2054–2064.

Shen, Y.; Ma, X.; Tan, Z.; Zhang, S.; Wang, W.; and 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*, 2782–2794. Association for Computational Linguistics.

Shen, Y.; Song, K.; Tan, X.; Li, D.; Lu, W.; and Zhuang, Y. 2023. DiffusionNER: Boundary Diffusion for Named Entity Recognition. In *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics*, 3875–3890. Toronto, Canada: Association for Computational Linguistics.

Souza, F.; Nogueira, R.; and Lotufo, R. 2019. Portuguese named entity recognition using BERT-CRF. *arXiv preprint arXiv:1909.10649*.

Vaswani, A.; Shazeer, N.; Parmar, N.; Uszkoreit, J.; Jones, L.; Gomez, A. N.; Kaiser, Ł.; and Polosukhin, I. 2017. Attention is all you need. *Advances in neural information processing systems*, 30.

Wang, S.; Sun, X.; Li, X.; Ouyang, R.; Wu, F.; Zhang, T.; Li, J.; and Wang, G. 2023. Gpt-ner: Named entity recognition via large language models. *arXiv preprint arXiv:2304.10428*.

Wang, X.; Jiang, Y.; Bach, N.; Wang, T.; Huang, Z.; Huang, F.; and Tu, K. 2021. Improving Named Entity Recognition by External Context Retrieving and Cooperative Learning. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics*, 1800–1812. Online: Association for Computational Linguistics.

Yamada, I.; Asai, A.; Shindo, H.; Takeda, H.; and 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)*, 6442–6454. Online: Association for Computational Linguistics.

Yan, H.; Gui, T.; Dai, J.; Guo, Q.; Zhang, Z.; and Qiu, X. 2021. A Unified Generative Framework for Various NER Subtasks. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics*, 5808–5822. Association for Computational Linguistics.

Yan, H.; Sun, Y.; Li, X.; and Qiu, X. 2022. An Embarrassingly Easy but Strong Baseline for Nested Named Entity Recognition. *arXiv preprint arXiv:2208.04534*.

Ying, C.; Cai, T.; Luo, S.; Zheng, S.; Ke, G.; He, D.; Shen, Y.; and Liu, T.-Y. 2021. Do transformers really perform badly for graph representation? *Advances in Neural Information Processing Systems*, 34: 28877–28888.

Yu, J.; Bohnet, B.; Poesio, M.; and Poesio, M. 2020. Named Entity Recognition as Dependency Parsing. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, 6470–6476. Association for Computational Linguistics.

Zhang, S.; Cheng, H.; Gao, J.; and Poon, H. 2023. Optimizing Bi-Encoder for Named Entity Recognition via Contrastive Learning. In *The Eleventh International Conference on Learning Representations*.

Zhong, Z.; and Chen, D. 2021. A Frustratingly Easy Approach for Entity and Relation Extraction. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, 50–61.

Zhou, K.; Li, Y.; and Li, Q. 2022. Distantly Supervised Named Entity Recognition via Confidence-Based Multi-Class Positive and Unlabeled Learning. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics*, 7198–7211.

Zhu, E.; and Li, J. 2022. Boundary Smoothing for Named Entity Recognition. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics*, 7096–7108. Association for Computational Linguistics.