DiFiNet: Boundary-Aware Semantic Differentiation and Filtration Network for Nested Named Entity Recognition

Yuxiang Cai[1], **Qiao Liu**[1*], **Yanglei Gan**[1], **Run Lin**[1], **Changlin Li**[1], **Xueyi Liu**[1], **Da Luo**[1], **Jiaye Yang**[1]
[1] University of Electronic Science and Technology of China
{yuxiangcai, yangleigan, runlin, changlinli, xueyiliu, luoda}@std.uestc.edu.cn,
edvincecilia@gmail.com, qliu@uestc.edu.cn

Abstract

Nested Named Entity Recognition (Nested NER) entails identifying and classifying entity spans within the text, including the detection of named entities that are embedded within external entities. Prior approaches primarily employ span-based techniques, utilizing the power of exhaustive searches to address the challenge of overlapping entities. Nonetheless, these methods often grapple with the absence of explicit guidance for boundary detection, resulting insensitivity in discerning minor variations within nested spans. To this end, we propose a Boundary-aware Semantic Differentiation and Filtration Network (DiFiNet) tailored for nested NER. Specifically, DiFiNet leverages a biaffine attention mechanism to generate a span representation matrix. This matrix undergoes further refinement through a self-adaptive semantic differentiation module, specifically engineered to discern semantic variances across spans. Furthermore, DiFiNet integrates a boundary filtration module, designed to mitigate the impact of non-entity noise by leveraging semantic relations among spans. Extensive experiments on three benchmark datasets demonstrate our model yields a new state-of-the-art performance[1].

1 Introduction

Named Entity Recognition (NER) involves the utilization of computer-assisted techniques to identify and extract entities and corresponding semantic types (Lample et al., 2016a), including *person* (PER), *location* (LOC), *geo-political entity* (GPE), and others. NER plays a crucial role in facilitating various downstream tasks such as relation extraction (Tang et al., 2022; Luo et al., 2024a), event extraction (Yang and Mitchell, 2016; Sha et al., 2018), and sentiment analysis (Liu et al., 2023).

Figure 1: A sample sentence from ACE Corpus containing nested entities.

Conventional approaches have primarily focused on identifying non-nested entities (Chiu and Nichols, 2016; Lample et al., 2016b; Ma and Hovy, 2016), a trend largely attributed to the constraints of corpus annotations that emphasize flat entity structures. However, the complex nature of natural language frequently features nested named entities, with studies revealing that approximately 30% of sentences in ACE04 and ACE05 datasets contain such structures (Finkel and Manning, 2009; Katiyar and Cardie, 2018). The prevalence of nested structures underscores the need for efficient models adept at handling such linguistic complexities.

In response to this challenge, recent years have witnessed a burgeoning interest in nested NER (Ju et al., 2018; Wang et al., 2020; Luo et al., 2024b). Among the emerging strategies, span-based models stand out as prominent approaches and have set new benchmarks in the field (Tan et al., 2020; Wang and Lu, 2020; Zhong and Chen, 2021; Zhu and Li, 2022). These models excel by leveraging exhaustive search techniques to systematically identify all possible spans, thereby capturing the full spectrum of nested structures.

Despite the success of span-based methods, they often struggle to fully utilize the rich semantics within spans due to the absence of explicit guidance for boundary detection. Previous research indicates that span-based models usually encounter confusion when dealing with nested entities characterized by a high degree of token overlap (Tan et al., 2021a; Zhu and Li, 2022; Wan et al., 2022). To illustrate, consider the sentence taken from the ACE05 dataset in Figure 1, entities like "*the*

*corresponding author
[1]The source code is available at: https://github.com/AONE-NLP/DiFiNet

Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers), pages 6455–6471
August 11-16, 2024

Alabama-Florida line", "*Florida*", and "*Alabama*", as well as non-entity spans such as "*Alabama-Florida line*" or "*the Alabama-Florida*" share a significant number of tokens, blurring the semantic distinction between entity and non-entity spans. Besides, those low-quality candidate spans, particularly long entities, incur significant computational costs (Tan et al., 2020; Wan et al., 2022) due to the extensive array of potential spans evaluated during training, inevitably limited by practical constraints.

To address this issue, we propose a Boundary-aware Semantic Differentiation and Filtration network (DiFiNet) explicitly incorporating semantic differences between nested spans as input features. By leveraging gradient back-propagation, DiFiNet learns appropriate internal representations to augment the distinction among nested entities within the span semantic representation space, enhancing its ability to discern boundaries and accurately classify nested named entities. Specifically, DiFiNet integrates BERT and a biaffine attention mechanism to construct a matrix of span semantic representations, followed by a self-adaptive semantic differentiation module to transform span representations into semantic differences across spans. Additionally, to alleviate the influence of low-quality candidate spans within the matrix, DiFiNet integrates a boundary filtration module. This module serves to model the interaction among spans, effectively reducing noise, with a specific focus on distinguishing semantically similar entity and non-entity spans. Our main contributions are summarized as follows:

- We tackle the challenge of nested named entity recognition from a novel perspective by explicitly enhancing boundary supervision to address the issue of boundary insensitivity within nested entities.
- Building upon our perspective, we propose a novel end-to-end framework which effectively captures subtle semantic variations between entity and non-entity spans. This framework is engineered to precisely detect entity boundaries via both self-adaptive semantic differentiation and boundary filtration module.
- Extensive experiments on the ACE04, ACE05, and GENIA datasets indicate that DiFiNet outperforms existing state-of-the-art models in the nested NER task. Further ablation studies validate the contribution of each module within our framework.

2 Related Work

Nested Named Entity Recognition is a task in Natural Language Processing (NLP) that involves identifying and classifying named entities within text data, where entities can have complex and overlapping structures. One approach to tackle this task is the hypergraph method, originally proposed by Lu and Roth (2015). This method maps the nested entity structures to sub-graphs in a hyper-graph and performs classification on them. Several extensions have been developed based on this method (Muis and Lu, 2017; Katiyar and Cardie, 2018).

Another approach is the hierarchical method introduced by Ju et al. (2018), which divides entities into different levels, where each deeper level represents a higher level of entity specificity. Following this paradigm, Wang et al. (2020) designed a pyramid sequence labeling framework using convolutional neural networks to extract entities from bottom to top. Shibuya and Hovy (2020) explored suboptimal path decoding to progressively extract entities hierarchically, and Wang et al. (2021) further improved it by excluding the influence of the optimal path. However, both hyper-graph and hierarchical methods suffer from high complexity when dealing with complex nested entities.

In contrast, Seq2Seq methods offer a simpler end-to-end approach, typically utilizing LSTM-CRF (Straková et al., 2019) or BART (Yan et al., 2021) to predict the label of each position. Zhang et al. (2022) improved Seq2Seq methods by adopting intra-entity and inter-entity de-confounding data augmentation techniques. Shen et al. (2023b) designed a dual-slot multi-prompt template with a position slot for locating and a type slot for typing, respectively. Nevertheless, when faced with highly complex nesting structures, these methods may encounter long-distance dependency problems, resulting in cascading errors.

To address the aforementioned challenges in nested NER, Sohrab and Miwa (2018) proposed a span-based method that treats the nested NER task as span prediction problems. This approach involves predicting potential entity spans for each token, followed by filtering and merging these spans to obtain the final nested entities. Building upon Sohrab's work, several works have made advancements to the span-based method by incorporating graph structure (Wan et al., 2022), valuable span patterns (Shen et al., 2021; Tan et al., 2021b) and attention mechanism (Yu et al., 2020; Xu et al.,

2021; Zheng et al., 2023) to achieve state-of-the-art performance. For example, Yu et al. (2020) proposed a biaffine attention mechanism to enhance the interaction between the start and end tokens and assigned scores to each span. When constructing the span-based contrastive loss function, Zhang et al. (2023a) utilizes concatenation to generate span representations. Shen et al. (2023a) redefined NER by modeling it as a boundary-denoising diffusion process. This approach generates named entities by refining and clarifying noisy spans.

However, span-based models typically utilize pooling (Eberts and Ulges, 2020; Shen et al., 2021; Li et al., 2021), concatenation (Li et al., 2021; Tan et al., 2020; Zheng et al., 2023) or integration (Zhu and Li, 2022; Yuan et al., 2022; Shen et al., 2023a) techniques to generate span representations from token representations. However, this approach often leads to generating semantically similar representations for highly overlapping spans. As a result, effectively capturing the subtle semantic nuances within individual spans becomes challenging.

To mitigate the boundary insensitivity issue, we propose to explicitly incorporate span semantic difference features into nested NER task. This allows the model to learn more robust span representations by capturing the nuanced semantic variations between entity and non-entity spans.

3 Our Approach

In this section, we introduce the details of our framework as shown in Figure 2. We first formulate the task definition of nested NER as follow,

Nested NER as boundary detection In the context of nested NER, the task involves analyzing an input sentence denoted as $X = \{x_1, x_2, \ldots, x_n\}$ to identify and classify potential entities according to a predefined set of entity types $T = \{t_1, t_2, \ldots, t_k\}$. Typically, an entity can be represented by a triplet (s_i, e_i, t_i), where s_i and e_i denote the starting and ending position of the entity, respectively, and $t_i \in T$ represents the assigned entity type. This structured representation allows for the precise localization of entity boundaries within the sentence. In a sentence with n tokens, there are a total of $n(n+1)/2$ valid spans.

3.1 Span Semantic Encoder

Given a sentence $X = \{x_1, x_2, \ldots, x_n\}$, we first utilize a pre-trained BERT model (Devlin et al., 2019) to vectorize each token x_i, resulting in token-level feature representations denoted as $\mathbf{H}_{enc} = \left\{\mathbf{h}_1, \mathbf{h}_2, \ldots, \mathbf{h}_n \mid \mathbf{h}_i \in \mathcal{R}^{h\times 1}\right\}$, where d is the embedding dimension, and n denotes the number of tokens within the sentence.

We then design two feed-forward neural networks (FNNs) to map the tokens and obtain the semantic representation vectors for the start and end tokens $\mathbf{h}_s, \mathbf{h}_e \in \mathcal{R}^{l\times h}$ of a span, where l represents the sentence length and h denotes the hidden dimension. Subsequently, a biaffine model is employed to combine the start and end token representation, and the width representation of the span ($\mathbf{w}_{ij} \in \mathcal{R}^{c\times 1}$) to construct the span representation matrix $\mathbf{M}^0 \in \mathcal{R}^{l\times l\times f}$, where f corresponds to the number of biaffine features.

For each span $\mathbf{S}_{ij}$, spanning from the i-th token to the j-th token, its vector $\mathbf{M}^0_{ij}$ is computed as:

$$\begin{aligned} \mathbf{h}_s &= \mathrm{GELU}\left(\mathbf{H}_{enc}\mathbf{W}_s\right); \mathbf{h}_e = \mathrm{GELU}\left(\mathbf{H}_{enc}\mathbf{W}_e\right), \\ \mathbf{M}^0_{ij} &= \left(\mathbf{h}_s[i] \oplus \mathbf{h}_e[j] \oplus \mathbf{w}_{ij}\right)\mathbf{W} + \mathbf{h}_s[i]\mathbf{U}\mathbf{h}_e[j]^T, \end{aligned} \quad (1)$$

where $\mathbf{W}_s, \mathbf{W}_e \in \mathcal{R}^{h\times h}$, $\mathbf{W} \in \mathcal{R}^{(2h+c)\times r}$, and $\mathbf{U} \in \mathcal{R}^{h\times r\times h}$ are learnable parameters. The feature size of biaffine model is denoted by r. $\oplus$ denotes concatenation, and GELU refers to the *gelu* activation function. It is worth noting that when $\mathbf{M}^0_{ij}$ is situated off the diagonal of the $\mathbf{M}^0$ matrix, the span representation $\mathbf{S}_{ij}$ exhibits two distinct forms, symmetrically arranged along the diagonal.

3.2 Self-adaptive Semantic Differentiation Module

To effectively capture the semantic differences between spans, we propose the **Self-adaptive Differentiation** operator (SAD), inspired by computer vision techniques such as the Roberts cross operator (Roberts and Lawrence, 1965). The SAD operator addresses the rigid nature of traditional gradient operators by adapting its differentiation template to the local semantic context of each span, bolstering the capability of handling subtle variations between semantically similar entity and non-entity spans.

The SAD operator functions in two primary phases: *the masking phase* and *the differentiation phase*. During *the masking phase*, a learnable convolutional kernel assesses local semantic regions, generating a mask matrix $\mathbf{mask}_{x_0}$ that highlights the most pertinent neighboring spans for semantic differentiation, formulated as:

$$I_{x_0} = \underset{i\in R\setminus\{x_0\}}{\arg\max}(\mathrm{LN}(\mathrm{Conv}(\mathbf{M}^0_{x_0}))), \quad (2)$$

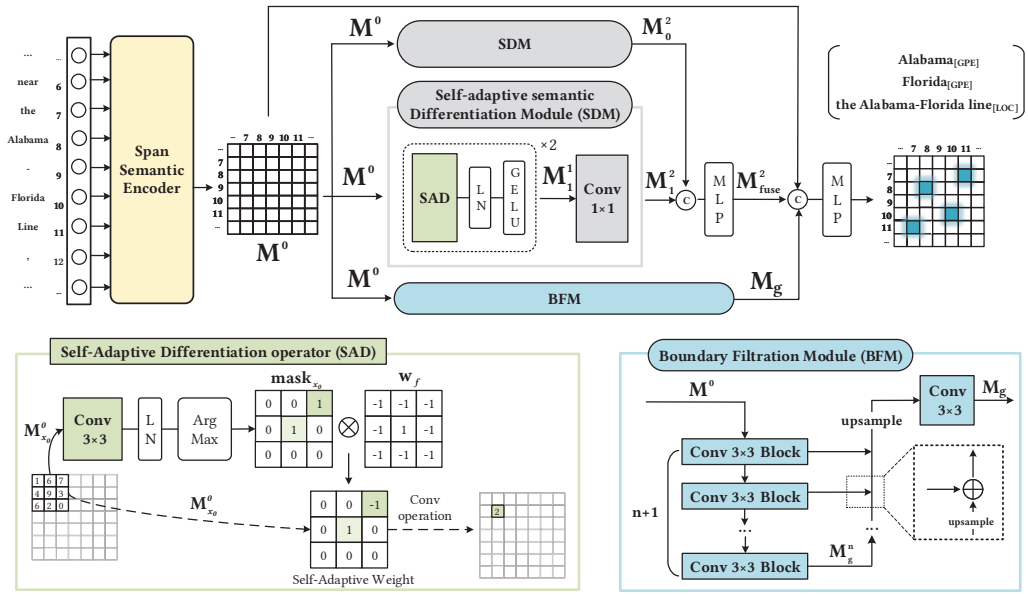

Figure 2: An overview of DiFiNet with two-layer structure SDM. ⓒ denoted the concatenate operation. $\oplus$ denoted the element-wise addition operation. $\otimes$ denoted Hadamard product operation. LN denotes LayerNorm layer.

$$\mathbf{mask}_{x_0}[i] = \begin{cases} 1 & if\ i = I_{x_0} or\ i = x_0 \\ 0 & others \end{cases}, \quad (3)$$

where x_0 denotes the position index of the convolution kernel center within the matrix M. $\mathrm{Conv}(\mathbf{M}^0_{x_0})$ represents the convolution operation on R centered around x_0, changing the channel number from f to the number of spans in R. LN denotes the layer normalization operation, and $\arg\max$ represents the position of the maximum score in R except for x_0.

The differentiation phase employs the mask matrix to apply self-adaptive weights to the span representations. This is achieved by element-wise multiplication of $\mathbf{mask}_{x_0}$ with a fixed weight matrix $\mathbf{w}_f$, enabling nuanced semantic differentiation tailored to each span's context:

$$\mathrm{SAD}(x_0) = \sum_{x_n \in R} \mathbf{mask}_{x_0} \cdot \mathbf{w}_f \cdot \mathbf{M}^0_{x_n}, \quad (4)$$

where $\mathbf{M}^0_{x_n}$ denotes the span semantic matrix at position x_n and $\mathbf{w}_f$ is the fixed weight matrix.

Integrated within the **Self-adaptive Semantic Differentiation Module** (**SDM**), the SAD operator underpins two SAD Blocks in each layer, designed to capture both first-order and second-order semantic differences between spans, denoted as:

$$\begin{aligned} &\mathrm{SADBlock}(*) = \mathrm{GELU}(\mathrm{LN}(\mathrm{SAD}(*))), \\ &\mathbf{M}^1_{lr} = \mathrm{SADBlock}(\mathrm{SADBlock}(\mathbf{M}^0)), \end{aligned} \quad (5)$$

where $lr \in \{0, 1, \ldots, N\}$ and $N+1$ is the number of layers in the model. For the sake of simplicity, the equations presented do not include the residual connections in SAD Blocks.

To enable back-propagation of gradients in the SAD operator, which contains a non-differentiable Argmax operation, we employ the Gumbel softmax estimator (Jang et al., 2016). Additionally, to ensure consistency in the differentiated objects, the fixed weights of the SAD operators in SDM have opposite signs. The weight matrix $\mathbf{w}_f$ is used for the first SAD operator, while the second SAD operator adopts the matrix $\mathbf{w}'_f$ whose elements are the negations of $\mathbf{w}_f$:

$$\mathrm{w}_f = \begin{bmatrix} -1 & -1 & -1 \\ -1 & 1 & -1 \\ -1 & -1 & -1 \end{bmatrix}, \mathbf{w}'_f = \begin{bmatrix} 1 & 1 & 1 \\ 1 & -1 & 1 \\ 1 & 1 & 1 \end{bmatrix}. \quad (6)$$

Subsequently, the SDM processes semantic difference features, aligning them within a standardized semantic framework. These processed features are then integrated using a linear layer, which consolidates the individual semantic distinctions into a comprehensive span boundary matrix $\mathbf{M}^2_{fuse}$:

$$\begin{aligned} \mathbf{M}^2_{lr} &= \mathrm{Conv}_{1\times1}(\mathbf{M}^1_{lr}), \\ \mathbf{M}^2_{fuse} &= \mathbf{W}_{fuse}(\underbrace{\mathbf{M}^2_0 \oplus \ldots \oplus \mathbf{M}^2_{lr}}_{N+1}) + \mathbf{B}_{fuse}, \end{aligned} \quad (7)$$

where $\mathbf{W}_{fuse}$ is the weight matrix and $\mathbf{B}_{fuse}$ is the bias term of the linear layer. The convolution operator $\mathrm{Conv}_{1\times1}$ denotes a 2D convolution operation with 1×1 kernel, and $\oplus$ indicates concatenation.

3.3 Boundary Filtration Module

The introduction of low-quality information, particularly in distinguishing non-entity spans, presents challenges in the SDM, potentially leading to an increase in false positives. To mitigate this, we introduce the **Boundary Filtration Module** (BFM), designed to reduce the impact of irrelevant span by extracting and utilizing interactions between spans, which aids in clarifying entity boundaries.

The BFM utilizes a structured methodology incorporating a top-down pathway for semantic interaction extraction and a bottom-up approach for detail restoration, complemented by lateral connections for comprehensive span relationship analysis. The top-down pathway employs a series of convolution blocks that apply Layer Normalization and the GELU activation function to refine span features systematically:

$$\begin{aligned} \mathrm{ConvBlock}(*) &= \mathrm{GELU}(\mathrm{LN}(\mathrm{Conv}(*))), \\ \mathbf{M}_g^0 &= \mathrm{ConvBlock}(\mathbf{M}^0), \\ \mathbf{M}_g^i &= \mathrm{ConvBlock}(\mathbf{M}_g^{i-1}), \end{aligned} \tag{8}$$

where $i \in \{1, 2, \ldots, n\}$ and $n + 1$ represents the number of convolution blocks.

The bottom-up pathway, in contrast, aims to restore finer details from higher-layer features through up-sampling, using nearest neighbor techniques to retain critical relational information. This is synchronized with lateral connections to merge features from different layers effectively, thereby avoiding loss of detail and preventing checkerboard artifacts typically associated with interpolation methods:

$$\begin{aligned} \mathbf{M}_g^{i-1'} &= \mathrm{upSample}(\mathbf{M}_g^i) + \mathbf{M}_g^{i-1}, \\ \mathbf{M}_g &= \mathrm{Conv}(\mathrm{upSample}(\mathbf{M}_g^{0'})). \end{aligned} \tag{9}$$

3.4 Span Semantic Decoder

In order to preserve the complete semantic information of span, we incorporate $\mathbf{M}^0$ as residual to $\mathbf{M}_{fuse}^2$. The composite matrix then undergoes linear decoding to yield prediction logits:

$$\mathbf{p} = \sigma(\mathbf{W}_p(\mathbf{M}^0 \oplus \mathbf{M}_{fuse}^2 \oplus \mathbf{M}_g) + \mathbf{B}_p), \tag{10}$$

where $\mathbf{p} \in \mathcal{R}^{l\times l\times t}$, $\mathbf{W}_p \in \mathcal{R}^{d\times t}$, $\mathbf{B}_p \in \mathcal{R}^t$. $\mathbf{W}_p$ and $\mathbf{B}_p$ are trainable parameters. σ denotes *Sigmoid* activation function.

3.5 Training and Inference

Training We minimize the following binary cross-entropy loss function:

$$\mathcal{L} = -\sum_{0\le i,j<l} \mathbf{y}_{ij}\log(\mathbf{p}_{ij}) + (1-\mathbf{y}_{ij})\log(1-\mathbf{p}_{ij}), \tag{11}$$

where $\mathbf{y}_{ij}$ is the ground truth entity type. To accommodate DiFiNet's architecture, which does not distinguish between the matrix halves during training, we incorporate errors from both the upper and lower triangular sections of $\mathbf{p}_{ij}$ and $\mathbf{p}_{ji}$, aligning with the symmetric nature of entity representation.

Inference For entity prediction, we average the values from the upper and lower sections of $\mathbf{p}$ to ensure consistent decoding:

$$\mathbf{p}'_{ij} = (\mathbf{p}_{ij} + \mathbf{p}_{ji})/2. \tag{12}$$

Following Yu et al. (2020), we first eliminate spans deemed non-entities (those with all probabilities below 0.5), then rank the remaining spans by their highest probability. Spans are selected sequentially; any span conflicting with previously chosen spans in terms of boundaries is omitted, maintaining clear entity demarcation.

4 Experiment

4.1 Datasets

We evaluate our model on three commonly used nested NER datasets: ACE04[2], ACE05 [3], and GENIA [4]. For the ACE datasets, we use the data preprocessing code released by Yan et al. (2023) and split the data into training, validation, and test sets by 8:1:1. For the GENIA dataset, we follow Li et al. (2022) to categorize entities into five types and split data into train, dev and test sets by 8:1:1. See Appendix A for detailed information of datasets.

4.2 Baselines

To evaluate the performance of the proposed model, we compare it with the following models on three datasets: Biaffine (Yu et al., 2020), Second-Best (Wang et al., 2021), Seq2Seq (Yan et al., 2021), Sequence2Set (Tan et al., 2021b), De-bias(Zhang

[2] https://catalog.ldc.upenn.edu/LDC2005T09

[3] https://catalog.ldc.upenn.edu/LDC2006T0

[4] http://www.geniaproject.org/genia-corpus

Table 1: The performance of various models on the ACE04, ACE05, and GENIA datasets is presented in Table 1. The "Encoder" column indicates the pre-trained models utilized by each model for the ACE datasets, while all models employed BioBERT-Base for the GENIA dataset. †signifies that the models were reproduced using the same pre-processed data and publicly available code. The best results are highlighted in **bold** font. The subscript denotes the standard deviation, providing a measure of result variability (e.g., 88.48_{23} indicates a value of 88.48±0.23).

Models	Encoder	ACE04			ACE05			GENIA		
		P	R	F1	P	R	F1	P	R	F1
Biaffine (2020)	BERT-base	87.30	86.00	86.70	85.20	85.60	85.40	81.8	79.30	80.50
Second-Best (2021)	BERT-base	86.42	85.71	86.06	83.95	84.67	84.30	79.20	78.16	78.63
Locate-and-Label (2021)	BERT-base	87.44	87.38	87.41	86.09	87.27	86.67	80.19	80.89	80.54
Seq2Seq (2021)	BART-large (2020)	87.27	86.41	86.84	83.16	86.38	84.74	78.57	79.30	78.93
Sequence2Set (2021b)	BERT-large	88.46	86.10	87.26	87.48	86.63	87.05	82.30	78.70	80.40
Span-Graph (2022)	BERT-base	86.70	85.93	86.31	84.37	85.87	85.11	77.92	80.74	79.30
De-bias (2022)	T5-base (2020)	86.36	84.54	85.44	83.31	86.56	84.90	81.04	77.21	79.08
BS (2022) †	RoBERTa-base	87.32	86.84	87.08	86.58	87.84	87.20	82.53	78.69	80.56
Triaffine (2022) †	BERT-large	87.13	87.68	87.40	86.70	86.94	86.82	80.42	**82.06**	81.23
W2NER (2022) †	BERT-large	87.19	87.72	87.45	85.77	87.80	86.76	83.10	79.76	81.39
ICR (2023)	BERT-large	-	-	-	87.11	87.14	87.12	79.02	80.68	79.87
BINDER(2023a) †	BERT-large	87.34	88.30	87.81	87.41	88.34	87.87	81.69	80.85	81.26
DiffusionNER(2023a) †	BERT-large	87.32	87.52	87.42	85.04	88.42	86.70	81.85	79.59	80.70
PromptNER(2023b) †	BERT-large	87.02	88.03	87.52	86.01	88.12	87.05	-	-	-
CNNNER (2023) †	RoBERTa-base	87.33	87.29	87.31	86.70	88.16	87.42	**83.19**	79.70	81.40
DiFiNet	RoBERTa-base	88.57	**88.43**	88.45_{14}	**89.16**	**88.74**	$\mathbf{88.94}_{38}$	83.01	80.80	$\mathbf{81.87}_{19}$
	BERT-large	**88.64**	88.32	$\mathbf{88.48}_{23}$	88.62	88.17	88.39_{31}			

et al., 2022), W2NER (Li et al., 2022), Locate-and-Label (Shen et al., 2021), BS (Zhu and Li, 2022), Triaffine(Yuan et al., 2022), Span-Graph (Wan et al., 2022), ICR(Zheng et al., 2023), BINDER (Zhang et al., 2023a), CNNNER (Yan et al., 2023), DiffusionNER(Shen et al., 2023a) and PromptNER(Shen et al., 2023b). See Appendix B and C for further elaboration on baseline models and implementation details of DiFiNet, respectively.

4.3 Main Results

Our evaluation employs three key metrics: **Precision**, **Recall**, and **F1-score**, to assess the performance of the models. We adopt strict evaluation criteria, whereby precise matches in both entity boundaries and categories are required for correct recognition. To validate the consistency and reliability of our findings, we conducted five separate trials, each initialized with distinct random seeds, and then proceeded to statistical analysis on the collected F1 scores. Specifically, we applied the T-test at a 5% significance level to determine the statistical significance of the differences observed between experimental outcomes.

Table 1 presents a comprehensive performance of DiFiNet and baseline models on ACE04, ACE05, and GENIA datasets for NER. Across all three NER datasets, DiFiNet consistently outperforms the baseline models. Notably, with RoBERTa-base as the underlying pre-trained model, DiFiNet secures an increase of +1.14% in F1-score on ACE04 and +1.52% in F1 on ACE05 compared to existing models. Similarly, when leveraging BERT-large as the pre-trained backbone, DiFiNet attains enhancements of +0.67% F1 on ACE04 and +0.52% F1 on ACE05. Additionally, DiFiNet exhibits an improvement of +0.47% F1 on the GENIA dataset. It is essential to highlight that the marginal gains on the GENIA dataset might stem from its significantly lower frequency of nested entities (18.41%) compared to ACE04 (45.68%) and ACE05 (39.11%), as shown in Table 7. These results underscore the superior performance of DiFiNet in addressing the complexities of nested NER.

4.4 Ablation Studies

Table 2 reports the F1 score results of the DiFiNet and its variants. The variations explored include disabling the Self-adaptive Semantic Differentiation Module (SDM) (**w/o SDM**), removing the Boundary Filtration Module (BFM) (**w/o BFM**), and excluding both (**w/o SDM, BFM**). Additionally, to gauge the impact of the self-adaptive mechanism, we examine a configuration without the self-adaptive mask (**w/o Self-adaptive mask**). Each variant demonstrates a drop in F1 score compared to DiFiNet model, highlighting the individual and collective importance of these modules in enhanc-

Table 2: Ablation experiment results (RoBERTa-base as the pre-trained language model). Δ denotes the performance drops (F1 Score) under different experimental conditions compared to our proposed model.

Settings	ACE04		ACE05	
	F1	Δ	F1	Δ
DiFiNet	**88.45**		**88.94**	
w/o SDM	87.43	-1.02	87.79	-1.15
w/o BFM	87.78	-0.67	88.21	-0.73
w/o SDM, BFM	87.09	-1.36	87.13	-1.81
w/o Self-adaptive mask	87.53	-0.92	88.32	-0.62
N = 0	87.60	-0.85	87.11	-1.83
N = 2	87.63	-0.82	88.01	-0.93
n = 0	87.48	-0.97	87.43	-1.51
n = 2	87.64	-0.81	88.24	-0.70
SAD Block × 1	87.59	-0.86	88.18	-0.76
SAD Block × 3	87.26	-1.19	87.84	-1.10

ing nested NER performance. Furthermore, we scrutinize the sensitivity of the model to various hyperparameters, such as the number of SDM layers, the number of SAD Blocks within each SDM layer, and the number of convolution blocks within the BFM. The adjustments are made while maintaining other settings at their optimal levels to isolate the effects of each parameter.

Our findings indicate the following: (1) **Necessity of SDM and BFM**: The elimination of either the SDM, the BFM, or both significantly diminishes the model's effectiveness. Such a reduction underscores the essential roles that these modules play in identifying semantic variances across spans and in bolstering the model's ability to detect boundaries; (2) **Adaptive Sampling Benefits**: Adaptive sampling within the differentiation process improves performance, indicating limitations in static approaches for handling complex nested entity structures; (3) **SDM Layer Impact**: Additional SDM layers do not guarantee improved outcomes, suggesting an optimal level of model complexity that avoids unnecessary noise; (4) **BFM Convolution Blocks**: Excessive convolution blocks in BFM don't lead to better results and may remove essential information, indicating a balance is needed; (5) **Optimization with SAD Blocks**: The model performs best with two SAD blocks, showing that this balance effectively captures semantic differences without overcomplicating the model. Overall, the experiments validate the importance of each proposed module in optimizing model performance.

Table 3: Results on CoNLL03 dataset. All models utilize BERT-large as a pretrain encoder, and all results are from their respective original papers.

Models	CoNLL03		
	P	R	F1
W2NER	92.71	93.44	92.07
DiffusionNER	92.99	92.56	92.78
PromptNER	92.48	92.33	92.41
BINDER	93.08	93.57	93.33
DiFiNet	**93.84**	**93.60**	**93.72**

4.5 Performance on Long Entities

Within NER tasks, the identification of long entities poses substantial challenges, notably due to a higher likelihood of encompassing nested structures, which exacerbates boundary insensitivity issues. Additionally, the accurate recognition of long entities represents a long-tail challenge (Wan et al., 2022), making their detection particularly complex.

In reflect the advantages of the DiFiNet model compared to other models in processing long entities, we have conducted our experimental comparison to include classic models from related work. Specifically, we have compared the performance of our proposed DiFiNet model with PromptNER, DiffusionNER, and CNNNER on the ACE04 and ACE05 datasets, shown in Table 4. All models were pretrained using RoBERTa-base as the language model. The results demonstrate that DiFiNet achieves state-of-the-art performance in recognizing entities of all lengths on the ACE05 dataset. On the ACE04 dataset, DiFiNet outperforms other models in all length ranges except for entities with lengths ranging from 13 to 16. Notably, DiFiNet shows significant absolute improvements of +10.94% and +16.17% in recognizing entities with lengths ranging from 10 to 13 and 13 to 16, respectively, on the ACE05 dataset. These improvements underscore DiFiNet's ability to discern subtle semantic variations among overlapping and extended spans, thereby enhancing its capability to identify the boundaries of complex entities.

4.6 Performance on Flat Entities

To evaluate the performance of our model on flat NER, we compared it against four leading state-of-the-art models on CoNLL03 dataset [5], W2NER (Li et al., 2022), DiffusionNER (Shen et al., 2023a), PromptNER (Shen et al., 2023b), and BINDER

[5] https://www.clips.uantwerpen.be/conll2003/ner/

Table 4: Entity length-wise results on ACE04 and ACE05 dataset. Entities are divided into six groups based on their lengths. The % column represents the proportion of entities in each length range out of the total number, rounded to two decimal places.

Datasets	Len	%	CNNNER	DiffusionNER	PromptNER	DiFiNet	Improvement (%)
ACE04	[1,4)	81.03	88.28	87.17	87.18	**90.34**	+1.96
	[4,7)	11.03	84.18	53.78	50.86	**84.92**	+0.74
	[7,10)	4.15	82.17	38.55	37.84	**84.37**	+2.20
	[10,13)	1.51	70.21	14.89	14.58	**73.91**	+3.70
	[13,16)	0.72	**77.27**	16.00	15.09	72.34	-4.93
	[16,+∞)	1.55	63.74	13.64	13.06	**66.67**	+2.93
ACE05	[1,4)	87.58	88.46	86.70	86.35	**89.92**	+0.46
	[4,7)	7.55	84.93	50.95	50.86	**86.04**	+1.11
	[7,10)	2.32	72.73	39.36	38.74	**80.68**	+7.95
	[10,13)	1.00	71.26	32.99	32.99	**82.22**	+10.94
	[13,16)	0.55	63.83	4.76	7.69	**80.00**	+16.17
	[16,+∞)	1.00	63.77	10.96	8.94	**67.50**	+3.78

Table 5: Runtime (seconds) comparison to variations and baselines.

Inference Time	ACE04	Δ	ACE05	Δ
DIFiNet	49	0	57	0
w/o SDM	47	-2	53	-4
w/o BFM	46	-3	55	-2
w/o BFM and SDM	42	-7	49	-8
DffsuionNER	193	144	253	196
PromptNER	172	123	215	158

(Zhang et al., 2023a). Table 3 shows that our model outperforms these benchmarks, particularly in precision metrics, achieving precision score of 93.84. This performance indicates that our model's explicit guidance for boundary detection not only aids nested entity recognition but also significantly enhances flat entity identification.

4.7 Inference Efficiency

Regarding the computational efficiency of DiFiNet, especially concerning the potential impact of integrating the Self-adaptive Semantic Differentiation Module (SDM) and the Boundary Filtration Module (BFM) on inference time, we conducted a comprehensive analysis of DiFiNet's performance in terms of inference efficiency, comparing it both with and without these modules, and against two state-of-the-art models, namely PromptNER and DiffusionNER, on the ACE04 and ACE05 datasets.

The experiment results is shown in Table 5. Our experiments, meticulously performed on an Nvidia A100 GPU with a batch size set to 1, reveal that the inclusion of the SDM and BFM modules introduces a negligible increase in computational overhead, with an observed increase in inference time of less than 10 seconds. Furthermore, the impact on inference efficiency when either the SDM or BFM is individually integrated is minimal, adding less than 5 seconds to the overall inference time.

More importantly, the analysis highlights that DiFiNet, even with the additional functionalities provided by the SDM and BFM, exhibits significantly better inference efficiency compared to PromptNER and DiffusionNER. Specifically, DiFiNet demonstrates a 3.51 to 3.93 times improvement in time efficiency over PromptNER and DiffusionNER on the ACE04 dataset, and a 3.77 to 4.43 times improvement on the ACE05 dataset. Our results strongly indicate that DiFiNet achieves an optimal balance between computational efficiency and model performance.

5 Case Study

Table 6 shows a case study conducted on ACE05 to compare DiFiNet with CNNNER (Yan et al., 2023). The first observation highlights that DiFiNet demonstrates superior ability to identify nested long entities due to its proficiency in detecting subtle distinctions between spans. The second sample demonstrates that DiFiNet excels in recognizing entities not encountered during training, leveraging semantic differences between spans. For instance, in the absence of training data for the boundary word "*fred*", it becomes challenging for the model to identify it based solely on span representation. However, by drawing guidance from the semantic difference between "*i am fred fred*" and "*fred fred*", DiFiNet can recognize the pattern of "*i am [name]*" in context, facilitating the accurate identification of the entity "*fred fred*". Furthermore, DiFiNet exhibits advantages in resolving ambiguous entity references. By leveraging the semantic difference between "*persuaded them otherwise*" and "*them*",

Table 6: Case Study on ACE05. The labels in the lower right corner indicate the entity type, while the superscripts indicate the nesting level. [The candidate entity]$_{\texttt{T}}$ denotes predicted with incorrect type T. {^{m1}The candidate entity^{m1}} represents the missed ground true entities whose number $m1$.

Sentence 1	
Ground True / DiFiNet	[^{1}These Iraqis^{1}]$_{\texttt{PER}}$ were rifling [^{1}a home of [^{2}a senior member of [^{3}the Mukhabarat^{3}]$_{\texttt{ORG}}$, [^{3}[^{4}Saddam^{4}]$_{\texttt{PER}}$' s dreaded secret police^{3}]$_{\texttt{ORG}}$ ^{2}]$_{\texttt{PER}}$ ^{1}]$_{\texttt{FAC}}$.
CNNNER	[^{1}These Iraqis^{1}]$_{\texttt{PER}}$ were rifling {^{m1}a home of {^{m2}a senior member of [^{1}the Mukhabarat^{1}]$_{\texttt{ORG}}$, [^{1}[^{2}Saddam^{2}]$_{\texttt{PER}}$' s dreaded secret police^{1}]$_{\texttt{ORG}}$ ^{m2}} ^{m1}}.
Sentence 2	
Ground True / DiFiNet	from [^{1}the [^{2}cnn^{2}]$_{\texttt{ORG}}$ center in [^{2}atlanta^{2}]$_{\texttt{GPE}}$ ^{1}]$_{\texttt{FAC}}$, [^{1}i^{1}]$_{\texttt{PER}}$' m [^{1}fred fred^{1}]$_{\texttt{PER}}$.
CNNNER	from [^{1}the [^{2}cnn^{2}]$_{\texttt{ORG}}$ center in [^{2}atlanta^{2}]$_{\texttt{GPE}}$ ^{1}]$_{\texttt{FAC}}$, [^{1}i^{1}]$_{\texttt{PER}}$' m {^{m1}fred fred^{m1}}.
Sentence 3	
Ground True / DiFiNet	But [^{1}neighboring Malaysia^{1}]$_{\texttt{GPE}}$' s success in integrating [^{1}[^{2}Russian^{2}]$_{\texttt{GPE}}$ MiG-29s^{1}]$_{\texttt{VEH}}$ and [^{1}[^{2}American^{2}]$_{\texttt{GPE}}$ [2F/A-18^{2}]$_{\texttt{VEH}}$ Hornets^{1}]$_{\texttt{VEH}}$ persuaded [^{1}them^{1}]$_{\texttt{PER}}$ otherwise, [Sudarsono]$_{\texttt{PER}}$ said.
CNNNER	But [^{1}neighboring Malaysia^{1}]$_{\texttt{GPE}}$' s success in integrating [^{1}[^{2}Russian^{2}]$_{\texttt{GPE}}$ MiG-29s^{1}]$_{\texttt{VEH}}$ and [^{1}[^{2}American^{2}]$_{\texttt{GPE}}$ [2F/A-18^{2}]$_{\texttt{VEH}}$ Hornets^{1}]$_{\texttt{VEH}}$ persuaded [^{1}them^{1}]$_{\texttt{GPE}}$ otherwise, [Sudarsono]$_{\texttt{PER}}$ said.

DiFiNet effectively recognizes the pattern of "*persuaded [person] otherwise*" and appropriately classifies "*them*" as PER. However, due to CNNNER lacking awareness of subtle semantic differences, it fails to correctly identify all entities in three examples. We provide extended case studies in Appendix D to further illustrate DiFiNet's ability to capture subtle semantic differences between spans.

6 Conclusion

This paper proposes a Boundary-aware Semantic Differentiation and Filtration Network (DiFiNet) to effectively address the issue of boundary insensitivity in nested named entity recognition tasks. DiFiNet introduces the self-adaptive semantic differentiation module to capture semantic difference information between spans and incorporates the boundary filtration module to reduce noise from non-entity spans and enhance the differences of boundary semantics between spans. Experimental results demonstrate that DiFiNet achieves superior performance compared to existing approaches on three benchmark datasets. Ablation experiments and case studies further validate the effectiveness of the proposed model. Looking ahead, we aim to extend utilization of boundary information in tasks such as event extraction and relation extraction.

Limitations

We discuss here the limitations of the method in this paper. First, this method still needs to traverse all spans, bringing high computational costs. Second, since the biaffine model encodes spans as continuous entities, it results in the prediction of only contiguous entities. Therefore, this method has limited applicability for noncontinuous entity recognition tasks. Finally, effectively integrating multi-level span semantic difference information is a promising direction for optimization.

Ethics Statement

To ensure ethical considerations, we will provide a detailed description as follows:

1. All of the datasets used are collected and annotated in previous studies. The use of these datasets in our work does not involve any interaction or collection of individual privacy data.

2. Our work focuses on methodology studies and experiments. The results and models in our paper will not be used to harm or deceive any individuals or groups.

3. There are no potential conflicts of interest or ethical issues regarding financial support in the sponsors and funds of our research work.

Acknowledgement

We would like to thank the anonymous reviewers for their valuable disucssion and constructive feedback. This work was supported by the National Natural Science Foundation of China (U22B2061, U2336204), the National Key R&D Program of

China (2022YFB4300603) and Sichuan Science and Technology Program (2023YFG0151).

References

Jason PC Chiu and Eric Nichols. 2016. Named entity recognition with bidirectional lstm-cnns. *Transactions of the association for computational linguistics*, 4:357–370.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 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, Volume 1 (Long and Short Papers)*, pages 4171–4186, Minneapolis, Minnesota. Association for Computational Linguistics.

Qingxiu Dong, Lei Li, Damai Dai, Ce Zheng, Zhiyong Wu, Baobao Chang, Xu Sun, Jingjing Xu, and Zhifang Sui. 2022. A survey on in-context learning. *arXiv preprint arXiv:2301.00234*.

Markus Eberts and Adrian Ulges. 2020. Span-based joint entity and relation extraction with transformer pre-training. In *the 24th European Conference on Artificial Intelligence*, pages 2006–2013, Online. IOS Press.

Jenny Rose Finkel and Christopher D Manning. 2009. Nested named entity recognition. In *Proceedings of the 2009 Conference on Empirical Methods in Natural Language Processing*, pages 141–150, Singapore. Association for Computational Linguistics.

Jiatao Gu, Daniel Jiwoong Im, and Victor O.K. Li. 2018. Neural machine translation with gumbel-greedy decoding. In *Proceedings of the Thirty-Second AAAI Conference on Artificial Intelligence and Thirtieth Innovative Applications of Artificial Intelligence Conference and Eighth AAAI Symposium on Educational Advances in Artificial Intelligence*, AAAI'18/IAAI'18/EAAI'18. AAAI Press.

Ridong Han, Tao Peng, Chaohao Yang, Benyou Wang, Lu Liu, and Xiang Wan. 2023. Is information extraction solved by chatgpt? an analysis of performance, evaluation criteria, robustness and errors. *arXiv preprint arXiv:2305.14450*.

Eric Jang, Shixiang Gu, and Ben Poole. 2016. Categorical reparameterization with gumbel-softmax. In *International Conference on Learning Representations*, pages 1–13, San Juan, Puerto Rico.

Meizhi Ju, Makoto Miwa, and Sophia Ananiadou. 2018. A neural layered model for nested named entity recognition. In *The 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, volume 1, pages 1446–1459, New Orleans, Louisiana. Association for Computational Linguistics.

Arzoo Katiyar and Claire Cardie. 2018. Nested named entity recognition revisited. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers)*, pages 861–871, New Orleans, Louisiana. Association for Computational Linguistics.

Takeshi Kojima, Shixiang (Shane) Gu, Machel Reid, Yutaka Matsuo, and Yusuke Iwasawa. 2022. Large language models are zero-shot reasoners. In *Advances in Neural Information Processing Systems*, volume 35, pages 22199–22213, New Orleans, USA. Curran Associates, Inc.

Guillaume Lample, Miguel Ballesteros, Sandeep Subramanian, Kazuya Kawakami, and Chris Dyer. 2016a. Neural architectures for named entity recognition. In *2016 Conference of the North American Chapter of the Association for Computational Linguistics*, pages 260–270, San Diego, USA. Association for Computational Linguistics.

Guillaume Lample, Miguel Ballesteros, Sandeep Subramanian, Kazuya Kawakami, and Chris Dyer. 2016b. Neural architectures for named entity recognition. In *Proceedings of the 2016 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 260–270.

Jinhyuk Lee, Wonjin Yoon, Sungdong Kim, Donghyeon Kim, Sunkyu Kim, Chan Ho So, and Jaewoo Kang. 2020. BioBERT: A pre-trained biomedical language representation model for biomedical text mining. *Bioinformatics (Oxford, England)*, 36(4):1234–1240.

Mike Lewis, Yinhan Liu, Naman Goyal, Marjan Ghazvininejad, Abdelrahman Mohamed, Omer Levy, Veselin Stoyanov, and Luke Zettlemoyer. 2020. BART: Denoising sequence-to-sequence pre-training for natural language generation, translation, and comprehension. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 7871–7880, Online. Association for Computational Linguistics.

Fei Li, ZhiChao Lin, Meishan Zhang, and Donghong Ji. 2021. A span-based model for joint overlapped and discontinuous named entity recognition. In *The 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing*, volume 1, pages 4814–4828, online. Association for Computational Linguistics.

Jingye Li, Hao Fei, Jiang Liu, Shengqiong Wu, Meishan Zhang, Chong Teng, Donghong Ji, and Fei Li. 2022. Unified named entity recognition as word-word relation classification. In *The AAAI Conference on Artificial Intelligence*, volume 36, pages 10965–10973, Washington, USA. AAAI Press.

Xueyi Liu, Rui Hou, Yanglei Gan, Da Luo, Changlin Li, Xiaojun Shi, and Qiao Liu. 2023. Aspect-oriented

opinion alignment network for aspect-based sentiment classification. In *Proceedings of the 26th European Conference on Artificial Intelligence (ECAI)*, pages 1552–1559, Kraków, Poland.

Yinhan Liu, Myle Ott, Naman Goyal, Jingfei Du, Mandar Joshi, Danqi Chen, Omer Levy, Mike Lewis, Luke Zettlemoyer, and Veselin Stoyanov. 2019. Roberta: A robustly optimized bert pretraining approach. *arXiv preprint arXiv:1907.11692*.

Ilya Loshchilov and Frank Hutter. 2019. Decoupled weight decay regularization. In *International Conference on Learning Representations*, pages 1–19, New Orleans, USA. OpenReview.net.

Jinghui Lu, Ziwei Yang, Yanjie Wang, Xuejing Liu, and Can Huang. 2024. Padellm-ner: Parallel decoding in large language models for named entity recognition. *arXiv preprint arXiv:2402.04838*.

Wei Lu and Dan Roth. 2015. Joint mention extraction and classification with mention hypergraphs. In *The 2015 Conference on Empirical Methods in Natural Language Processing*, pages 857–867, Lisbon, Portugal. Association for Computational Linguistics.

Da Luo, Yanglei Gan, Rui Hou, Run Lin, Qiao Liu, Yuxiang Cai, and Wannian Gao. 2024a. Synergistic anchored contrastive pre-training for few-shot relation extraction. volume 38, pages 18742–18750, Vancouver, Canada.

Da Luo, Run Lin, Qiao Liu, Yuxiang Cai, Xueyi Liu, Yanglei Gan, and Rui Hou. 2024b. Synergetic interaction network with cross-task attention for joint relational triple extraction. In *Proceedings of the 2024 Joint International Conference on Computational Linguistics, Language Resources and Evaluation (LREC-COLING 2024)*, pages 15457–15468, Torino, Italia. ELRA and ICCL.

Xuezhe Ma and Eduard Hovy. 2016. End-to-end sequence labeling via bi-directional LSTM-CNNs-CRF. In *The 54th Annual Meeting of the Association for Computational Linguistics*, volume 1, pages 1064–1074, Berlin, Germany. Association for Computational Linguistics.

Aldrian Obaja Muis and Wei Lu. 2017. Labeling gaps between words: Recognizing overlapping mentions with mention separators. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*, pages 2608–2618, Copenhagen, Denmark. Association for Computational Linguistics.

Razvan Pascanu, Tomas Mikolov, and Yoshua Bengio. 2013. On the difficulty of training recurrent neural networks. In *International Conference on Machine Learning*, pages 1310–1318, Atlanta, USA. Pmlr, PMLR.

Colin Raffel, Noam Shazeer, Adam Roberts, Katherine Lee, Sharan Narang, Michael Matena, Yanqi Zhou, Wei Li, and Peter J. Liu. 2020. Exploring the limits of transfer learning with a unified text-to-text transformer. *Journal of Machine Learning Research*, 21(1):1–67.

Roberts and G. Lawrence. 1965. *Machine Perception of Three-Dimensional Solids*. Thesis, MIT: Massachusetts Institute of Technology.

Lei Sha, Feng Qian, Baobao Chang, and Zhifang Sui. 2018. Jointly extracting event triggers and arguments by dependency-bridge RNN and tensor-based argument interaction. In *The Thirty-Second AAAI Conference on Artificial Intelligence and Thirtieth Innovative Applications of Artificial Intelligence Conference and Eighth AAAI Symposium on Educational Advances in Artificial Intelligence*, volume 32, pages 5916–5923, Louisiana, USA. AAAI Press.

Yongliang Shen, Xinyin Ma, Zeqi Tan, Shuai Zhang, Wen Wang, and Weiming Lu. 2021. Locate and label: A two-stage identifier for nested named entity recognition. In *The 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing*, volume 1, pages 2782–2794, Online. Association for Computational Linguistics.

Yongliang Shen, Kaitao Song, Xu Tan, Dongsheng Li, Weiming Lu, and Yueting Zhuang. 2023a. DiffusionNER: Boundary diffusion for named entity recognition. In *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 3875–3890, Toronto, Canada. Association for Computational Linguistics.

Yongliang Shen, Zeqi Tan, Shuhui Wu, Wenqi Zhang, Rongsheng Zhang, Yadong Xi, Weiming Lu, and Yueting Zhuang. 2023b. PromptNER: Prompt locating and typing for named entity recognition. In *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 12492–12507, Toronto, Canada. Association for Computational Linguistics.

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

Mohammad Golam Sohrab and Makoto Miwa. 2018. Deep exhaustive model for nested named entity recognition. In *The Conference on Empirical Methods in Natural Language Processing*, pages 2843–2849, Brussels, Belgium. Association for Computational Linguistics.

Jana Straková, Milan Straka, and Jan Hajic. 2019. Neural architectures for nested NER through linearization. In *The 57th Annual Meeting of the Association for Computational Linguistics*, pages 5326–5331, Florence, Italy. Association for Computational Linguistics.

Chuanqi Tan, Wei Qiu, Mosha Chen, Rui Wang, and Fei Huang. 2020. Boundary enhanced neural span

classification for nested named entity recognition. In *The Thirty-Fourth AAAI Conference on Artificial Intelligence*, volume 34, pages 9016–9023, New York, USA. AAAI Press.

Zeqi Tan, Yongliang Shen, Shuai Zhang, Weiming Lu, and Yueting Zhuang. 2021a. A sequence-to-set network for nested named entity recognition. In *Proceedings of the Thirtieth International Joint Conference on Artificial Intelligence, IJCAI-21*, pages 3936–3942. International Joint Conferences on Artificial Intelligence Organization. Main Track.

Zeqi Tan, Yongliang Shen, Shuai Zhang, Weiming Lu, and Yueting Zhuang. 2021b. A sequence-to-set network for nested named entity recognition. In *The Thirtieth International Joint Conference on Artificial Intelligence, IJCAI-21*, pages 3936–3942, Online. International Joint Conferences on Artificial Intelligence Organization.

Wei Tang, Benfeng Xu, Yuyue Zhao, Zhendong Mao, Yifeng Liu, Yong Liao, and Haiyong Xie. 2022. UniRel: Unified representation and interaction for joint relational triple extraction. In *Proceedings of the 2022 Conference on Empirical Methods in Natural Language Processing*, pages 7087–7099, Abu Dhabi, United Arab Emirates. Association for Computational Linguistics.

Juncheng Wan, Dongyu Ru, Weinan Zhang, and Yong Yu. 2022. Nested named entity recognition with span-level graphs. In *The 60th Annual Meeting of the Association for Computational Linguistics*, volume 1, pages 892–903, dublin, Ireland. Association for Computational Linguistics.

Jue Wang and Wei Lu. 2020. Two are better than one: Joint entity and relation extraction with table-sequence encoders. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 1706–1721, Online. Association for Computational Linguistics.

Jue Wang, Lidan Shou, Ke Chen, and Gang Chen. 2020. Pyramid: A layered model for nested named entity recognition. In *The 58th Annual Meeting of the Association for Computational Linguistics*, pages 5918–5928, Online. Association for Computational Linguistics.

Yiran Wang, Hiroyuki Shindo, Yuji Matsumoto, and Taro Watanabe. 2021. Nested named entity recognition via explicitly excluding the influence of the best path. In *The 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing*, pages 3547–3557, Online. Association for Computational Linguistics.

Jingjing Xu, Xu Sun, Zhiyuan Zhang, Guangxiang Zhao, and Junyang Lin. 2019. *Understanding and improving layer normalization*. Curran Associates Inc., Red Hook, NY, USA.

Yongxiu Xu, Heyan Huang, Chong Feng, and Yue Hu. 2021. A supervised multi-head self-attention network for nested named entity recognition. In *Thirty-Fifth AAAI Conference on Artificial Intelligence, AAAI 2021, Thirty-Third Conference on Innovative Applications of Artificial Intelligence, IAAI 2021, The Eleventh Symposium on Educational Advances in Artificial Intelligence, EAAI 2021*, pages 14185–14193, Online. AAAI Press.

Hang Yan, Tao Gui, Junqi Dai, Qipeng Guo, Zheng Zhang, and Xipeng Qiu. 2021. A unified generative framework for various NER subtasks. In *The 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing*, pages 5808–5822, Online. Association for Computational Linguistics.

Hang Yan, Yu Sun, Xiaonan Li, and Xipeng Qiu. 2023. An embarrassingly easy but strong baseline for nested named entity recognition. In *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pages 1442–1452, Toronto, Canada. Association for Computational Linguistics.

Bishan Yang and Tom Mitchell. 2016. Joint extraction of events and entities within a document context. In *The 15th Annual Conference of the North American Chapter of the Association for Computational Linguistics*, pages 289–299, San Diego, USA. Association for Computational Linguistics.

Juntao Yu, Bernd Bohnet, and Massimo Poesio. 2020. Named entity recognition as dependency parsing. In *The 58th Annual Meeting of the Association for Computational Linguistics*, pages 6470–6476, Online. Association for Computational Linguistics.

Zheng Yuan, Chuanqi Tan, Songfang Huang, and Fei Huang. 2022. Fusing heterogeneous factors with triaffine mechanism for nested named entity recognition. In *Findings of the Association for Computational Linguistics: ACL 2022*, pages 3174–3186, dublin, Ireland. Association for Computational Linguistics.

Sheng Zhang, Hao Cheng, Jianfeng Gao, and Hoifung Poon. 2023a. Optimizing bi-encoder for named entity recognition via contrastive learning. In *The Eleventh International Conference on Learning Representations*.

Shengyu Zhang, Linfeng Dong, Xiaoya Li, Sen Zhang, Xiaofei Sun, Shuhe Wang, Jiwei Li, Runyi Hu, Tianwei Zhang, Fei Wu, et al. 2023b. Instruction tuning for large language models: A survey. *arXiv preprint arXiv:2308.10792*.

Shuai Zhang, Yongliang Shen, Zeqi Tan, Yiquan Wu, and Weiming Lu. 2022. De-bias for generative extraction in unified NER task. In *The 60th Annual Meeting of the Association for Computational Linguistics*, pages 808–818, dublin, Ireland. Association for Computational Linguistics.

Qinghua Zheng, Yuefei Wu, Guangtao Wang, Yanping Chen, Wei Wu, Zai Zhang, Bin Shi, and Bo Dong. 2023. Exploring interactive and contrastive relations for nested named entity recognition. *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, 31:2899–2909.

Zexuan Zhong and Danqi Chen. 2021. A frustratingly easy approach for entity and relation extraction. In *2021 Annual Conference of the North American Chapter of the Association for Computational Linguistics*, pages 50–61, Online. Association for Computational Linguistics.

Enwei Zhu and Jinpeng Li. 2022. Boundary smoothing for named entity recognition. In *The 60th Annual Meeting of the Association for Computational Linguistics*, pages 7096–7108, dublin, Ireland. Association for Computational Linguistics.

Table 7: Statistics of the datasets used in the experiments. The "Len" column represents the average length of sentences or entities in each dataset.

		Train	Dev	Test	Len	Overlap rate
ACE04	Sen	6,297	742	824	23.52	45.68%
	Ent	22,231	2,514	3,036	2.64	
ACE05	Sen	7,178	960	1,051	20.59	39.11%
	Ent	25,300	3,321	3,099	2.40	
GENIA	Sen	15,023	1,669	1,854	25.41	18.41%
	Ent	45,144	5,365	5,506	1.97	

A Data Statistics

The ACE04 and ACE05 datasets contain seven entity types: Person (`PER`), Organization (`ORG`), Geo-Political Entity (`GPE`), Location (`LOC`), Facility (`FAC`), Weapon (`WEA`), and Vehicle (`VEH`). The GENIA datasets including five categories: `DNA`, `RNA`, `Protein`, `Cell line`, and `Cell type`. According to statistical analysis, 30% of the sentences in the ACE04 and ACE05 datasets contain nested entities, while the GENIA dataset has 17% of sentences with nested entities. The statistical information of the three benchmark datasets is shown in Table 7.

It is worth emphasizing that Yan et al. (2023) observed that despite the usage of the same dataset in recent studies (Wan et al., 2022; Zhu and Li, 2022; Yuan et al., 2022; Li et al., 2022), the statistics of the training datasets differ due to variations in preprocessing methods. Consequently, it would be unfair to directly compare model performance using different versions. In order to address this concern, we utilized the preprocessing code provided by(Yan et al., 2023) and applied it to our dataset. Subsequently, we re-implemented several baseline models in 2022 using the preprocessed dataset and publicly available code. The performance metrics of these models are recorded in Table 1. However, due to the unavailability of code and limited model details, we were unable to fully replicate the Span Graph(Wan et al., 2022) and De-bias(Zhang et al., 2022) models.

B Baseline Details

We compare our method with the following baselines:

1) Biaffine: Yu et al. (2020)used a biaffine model to identify nested named entities, predicting the named entity boundaries by predicting the dependency relationship between two words.

2) Second-Best: Wang et al. (2021) recognized

nested entities by explicitly excluding the influence of the optimal path of the probability graph.

3) Seq2Seq: Yan et al. (2021) used a pointer-based approach to convert the entity tagging task into a sequence generation task.

4) Sequence2Set: Tan et al. (2021b) proposed a novel neural network architecture for set prediction specifically for nested NER.

5) De-bias: Zhang et al. (2022) analyzed the incorrect biases in the generation process and used the intra- and inter-entity de-confounding data augmentation methods, to reduce the model's bias.

6) W2NER: Li et al. (2022) modeled unified NER as word-word relationship classification, avoiding conflicts between labels in traditional sequence labeling methods.

7) Locate-and-Label: Shen et al. (2021) modeled the nested NER task as a joint task of entity boundary regression and span classification, improving the training and inference efficiency.

8) BS: Zhu and Li (2022) proposed a boundary smoothing method, which reassigns probabilities from annotated spans to the surrounding ones, to improve the performance of NER models.

9) CNNNER: Yan et al. (2023) used CNN to model the spatial relationships in the score matrix to solve the nested named entity recognition task.

10) Triaffine: Yuan et al. (2022) improved entity recognition performance by obtaining various interaction information between heterogeneous elements such as tokens, entity types, and boundaries.

11) Sequence2Set: Tan et al. (2021b) proposed a novel neural network architecture for set prediction specifically for nested NER.

12) Span-Graph: Wan et al. (2022) modeled nested NER using a span-based graph structure, where each span is represented as a node and spans are connected by edges to enhance the semantic representation capability of the spans.

13) DiffusionNER: Shen et al. (2023a) used the diffusion model for NER task, generating entities by progressive boundary refinement over the noisy spans.

14) PromptNER: Shen et al. (2023b) designs a dual-slot multi-prompt template with the position slot and type slot to prompt locating and typing respectively.

15) DINDER: Zhang et al. (2023a) frame NER as a representation learning problem that maximizes the similarity between the vector representations of entity mentions and their types.

Table 8: Hyper-parameter settings on different benchmarks

	ACE04	ACE05	GENIA
Batchsize	48	48	8
Epoch	80	80	10
Learning rate	2e-5	2e-5	7e-6
Biaffine size	120	120	400
CNN channel dim	120	120	200
Dropout rate	0.2	0.2	0.1

16) ICR: Zheng et al. (2023) introduces a scale transformation mechanism and a supervised contrastive learning loss to explore interactive and contrastive relations among spans.

C Hyper-parameter Details

We utilize RoBERTa-Base (Liu et al., 2019) and BERT-Large (Devlin et al., 2019) as the pre-trained models for the ACE dataset, with a hidden layer size of 768. For the GENIA dataset, we employ BioBERT-Base-v1.1 (Lee et al., 2020) as the pre-trained model, also with a hidden layer size of 768. In the SDM module, the number of layers $N + 1$ is set to 2 for all datasets. In the BFM module, the number of convolution blocks $n + 1$ is set to 2 for all datasets. Except for the extra annotation, the size of Conv used in the model is 3×3. To minimize memory usage, the SAD operator employed in each layer of the SDM shares parameters, except having different fixed weight templates. The hyper-parameters for the biaffine model were chosen based on the study conducted by (Yan et al., 2023), which also incorporates the multi-head biaffine attention mechanism in its implementation. We set the number of heads to 4 and introduce a span width embedding with a size of 25. By default, the temperature parameter in the Gumbel Softmax estimator is set to 1.

Our model is trained using the AdamW optimizer (Loshchilov and Hutter, 2019). To control overfitting, the L2 norm of the gradient is limited to within 5 by gradient clipping (Pascanu et al., 2013), employed by our model. In the first 10% of the training steps, we gradually increased the learning rate using a linear warm-up scheduler. After the warm-up period, we gradually reduced the learning rate using a linear decay scheduler. All experiments are conducted on NVIDIA Tesla A100 (80G). Other hyper-parameters that vary depending on the datasets are detailed in Table 8.

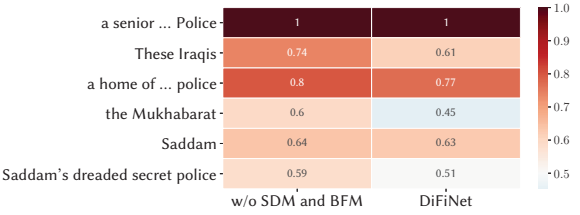

Figure 3: illustrations of the semantic similarity heatmaps for the entity "*a senior member of the Mukhabarat, Saddam's dreaded secret police*" in Sentence 1. The heatmaps compare two cases: "w/o SDM and BFM" (without Semantic Difference Modeling and Boundary Fusion Module) and "DiFiNet" (with SDM and BFM).

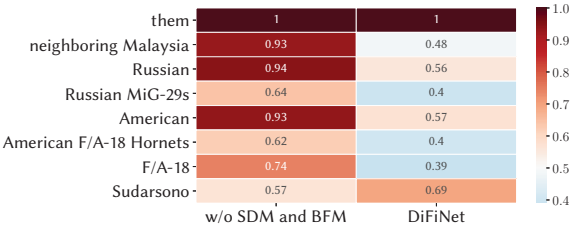

Figure 4: Semantic similarity visualization. It illustrates the semantic similarity between the entity "*them*" and other entities in Sentence 3. Each column has a similar meaning to the corresponding column in Figure 3.

D Semantic Similarity Visualization and Analysis of Cases

In order to visualize the semantic similarity between example instances from Table 6, we provide corresponding semantic similarity heatmaps. Specifically, Figure 3 and Figure 4 display partial semantic similarity heatmaps for instance 1 and instance 3, respectively. To ensure optimal clarity, we present the complete semantic similarity heatmap for instance 2, as shown in Figure 5 and Figure 6. Within these heatmaps, each value within a color block represents the cosine similarity of the span semantic vectors.

Figure 3 demonstrates the effectiveness of appropriately modeling semantic difference information between spans in addressing the issue of boundary insensitivity between nested entities. By utilizing SDM and BFM, the semantic similarity between different nested entities decreases, facilitating their differentiation by the classifier. For instance, in Figure 3, the entity "*a senior ... police*" with PER type exhibits a 15% decrease in semantic similarity with the entity "*the Mukhabarat*" of ORG type. In Figure 4, the explicit incorporation of span semantic difference information is shown to enhance

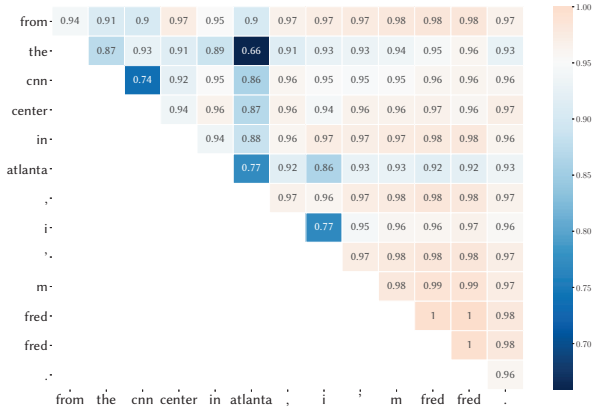

Figure 5: The similarity heatmap of span semantics generated by DiFiNet without SDM and BFM. It illustrates the semantic similarity between the entity "*fred fred*" and other entities in Sentence 1. The tokens on the vertical axis represent the starting tokens of the spans, while the tokens on the horizontal axis represent the ending tokens of the spans.

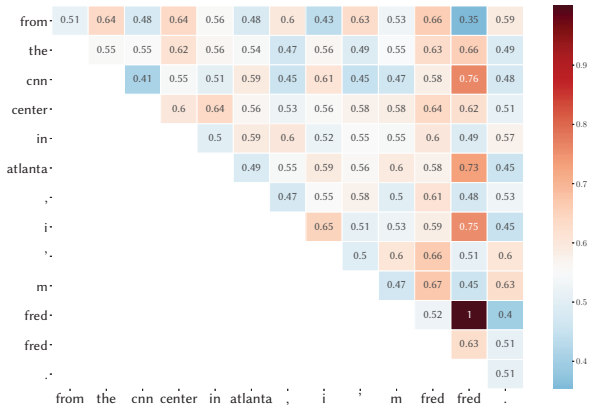

Figure 6: Display of the similarity heatmap of span semantics generated by DiFiNet, with the same vertical and horizontal axis settings as Figure 5.

the semantic representation capability of DiFiNet, leading to improved overall robustness. In the absence of SDM and BFM, the similarity between "*them*" and other entities tends to be relatively high, with over half of the entities displaying a similarity of 70% or higher. However, with the inclusion of semantic difference information, the similarity between entities decreases significantly. Even the highest semantic similarity, which occurs with the same type entity "*sudarsono*", remains below 70%.

The visualization results of sentence 2 (Figure 5 and 6) further validate the aforementioned observation. The original model faces difficulties in distinguishing the named entity "*fred fred*" from non-entity spans when confronted with the unseen boundary word "*fred*", leading to a boundary insen-

Table 9: Comparsion experiments with LLMs

Models	Setting	ACE04	ACE05	Genia	CoNLL03
ChatGPT	0-shot	27.80	23.38	38.09	60.10
ChatGPT	5-shot + ICL	38.52	36.17	48.82	70.53
ChatGPT	5-shot +CoT	40.57	33.98	50.89	74.63
PADeLLM-NER	Instruction Tuning	——	85.02	77.66	92.52
DiFiNet	——	88.45	**88.94**	**81.87**	**93.72**

sitivity problem. In contrast, by leveraging the guidance of span semantic difference information, the model accurately identifies "*fred fred*" as a named entity of type `PER` and successfully distinguishes it from the nearly identical span "*fred*".

E Comparsion with GPT Models

LLMs demonstrate the remarkable capabilities across a spectrum of NLP tasks. To futher experiments, we incorporated these LLMs as benchmarks to validate the efficacy of DiFiNet in handling the complexities inherent in Nested NER.

Our comparative analysis, incorporating data from recent studies (Han et al., 2023; Lu et al., 2024) in Table 9, reveals that under a zero-shot setting, GPT-3.5's performance significantly falls short when compared to DiFiNet. This trend persists even under a five-shot setting, despite employing strategies such as In-Context Learning (ICL) prompts (Dong et al., 2022) and the Chain of Thought (CoT) reasoning (Kojima et al., 2022), indicating that GPT-3.5 struggles to match DiFiNet's performance in Nested NER tasks.

Furthermore, we evaluated the PaDeLLM-NER, a version of Llama2 specifically fine-tuned for the NER task, using the ACE05, GENIA, and CoNLL03 datasets. Despite the Llama2-7b model's comprehensive tuning (Zhang et al., 2023b), DiFiNet demonstrated superior performance across all mentioned datasets, outperforming PaDeLLM-NER by noticeable margins. DiFiNet outperforms PaDeLLM-NER in terms of F1-score by 3.92% and 4.21% on the ACE05 and GENIA datasets, respectively. Additionally, on the flat NER dataset CoNLL03, DiFiNet's performance also surpasses that of PaDeLLM-NER by 1.20%.

Above results suggest that in the nested NER domain, DiFiNet significantly outperforms the aforementioned LLMs while maintaining substantially fewer model parameters. These outcomes underscore DiFiNet's exceptional capability in accurately identifying and distinguishing nested entities, a task where LLMs, due to their potential oversensitivity to irrelevant context (Han et al., 2023), may not perform as effectively. DiFiNet's architecture, designed to discern subtle semantic differences and filter out non-entity spans, affords it a distinct advantage in processing complex sentence structures and accurately identifying multiple nested spans.

F The Stability Test for Gumbel Softmax Operation

The Gumbel-Softmax reparameterization technique is indeed crucial for enabling gradient backpropagation through discrete variables by providing a differentiable approximation of the argmax function, thereby maintaining the differentiability of the SAD operator in scenarios that traditionally rely on non-differentiable operations.

To elaborate, the Gumbel-Softmax operation is defined by the equation:

$$z_i = \frac{\exp\left(\left(\log\left(\pi_i\right) + g_i\right)/\tau\right)}{\sum_{j=1}^{K} \exp\left(\left(\log\left(\pi_j\right) + g_j\right)/\tau\right)} \tag{13}$$

where, $z_i$represents the output for category i, g_i is a noise term sampled from the Gumbel distribution, π_i is the log probability of the original input, and K is the total number of categories. As indicated in (Jang et al., 2016), a smaller value of τ results in the distribution of being closer to one-hot encoding, albeit with larger gradient variances. Conversely, a larger τ yields a smoother distribution of z with smaller gradient variances.

However, the potential training stability issues may arise from the choice of τ during training within the Gumbel-Softmax operation (Gu et al., 2018). To test the training stability of our model, we carefully conducted additional sensitivity analysis on the temperature coefficient τ by setting the τ range from 0.1 to 0.9.

As illustrated in Table 10, our results, underpinned by statistical analysis, indicate that the model's effectiveness is maintained across diverse

Table 10: Sensitivity analysis on the temperature coefficient τ.

τ	ACE04			ACE05		
	P	R	F1	P	R	F1
0.10	88.08	89.10	88.59	88.43	89.25	88.84
0.30	88.02	89.13	88.57	88.19	90.09	89.13
0.50	88.70	88.61	88.65	89.11	88.48	88.80
0.70	88.15	88.61	88.38	88.37	89.00	88.68
0.90	87.81	88.77	88.29	88.02	89.13	88.57
Average	88.15	88.84	88.50	88.42	89.19	88.80

settings, with no notable decline in essential performance metrics across all evaluated datasets. This consistency is attributed to the implementation of a Gumbel-Softmax operation followed by a LayerNorm layer, which recalibrates and rescales the backward gradients to maintain distribution stability (Xu et al., 2019).