Mitigating Prototype Shift: Few-Shot Nested Named Entity Recognition with Prototype-Attention

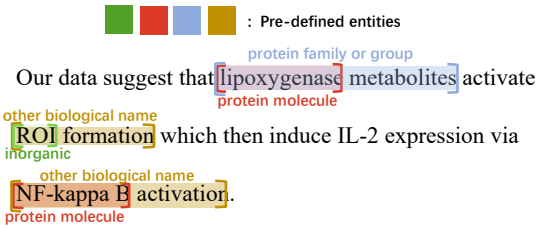
Anonymous ACL submission

Abstract

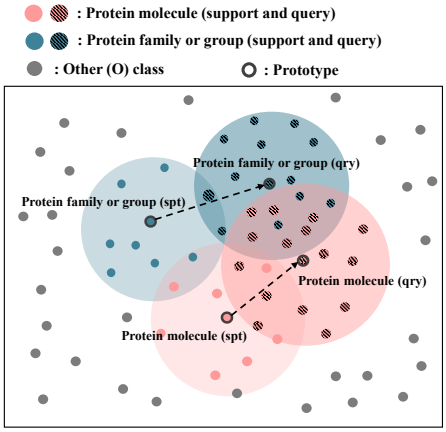
Nested entities are prone to obtain similar representations in pre-trained language models, posing challenges for Named Entity Recognition (NER), especially in the few-shot setting where prototype shifts often occur due to distribution differences between the support and query sets. In this paper, we regard entity representation as the combination of prototype and non-prototype representations. With a hypothesis that using the prototype representation specifically can help mitigate potential prototype shifts, we propose a Prototype-Attention mechanism in the Contrastive Learning framework (PACL) for the few-shot nested NER. PACL first generates prototype-enhanced span representations to mitigate the prototype shift by applying a prototype attention mechanism. It then adopts a novel prototype-span contrastive loss to reduce prototype differences further and overcome the O-type's non-unique prototype limitation by comparing prototype-enhanced span representations with prototypes and original semantic representations. Our experiments on three English, German, and Russian nested NER datasets show that the PACL outperformed seven baseline models on the 1-shot and 5-shot tasks in terms of F_1 score. Further analyses indicate that our Prototype-Attention mechanism has high generality, enhancing the performance of two baseline models, and can serve as a valuable tool for NLP practitioners facing few-shot nested NER tasks.

1 Introduction

The few-shot Named Entity Recognition (NER) task has become increasingly appealing because traditional NER methods rely on many labeled training instances that are both time-consuming and experience-dependent. It aims to deal with the NER task with the support of only very few labeled instances. Researchers have made significant progress on this task by applying deep learning models, including pre-trained-model-based (Florez and Mueller, 2019; Hou et al., 2019; Yang et al., 2021; Wang et al., 2022b), metric-learning-based (Snell et al., 2017; Hofer et al., 2018; Yang and Katiyar, 2020), meta-learning-based (Li et al., 2020a; Sung et al., 2018), prompt-tuning-based (Ma et al., 2022; Hou et al., 2022), and contrastive-learning-based (Das et al., 2022).



(a) Nested entities



(b) Prototype shift

Figure 1: **(a)** Example of an instance with nested entities from the GENIA dataset. **(b)** Illustration of prototype shifts, where the prototypes differ due to the distribution difference between the support and query sets.

While most above research focus on flat entities that don't overlap, the few-shot nested NER task has started gaining attention (Ming et al., 2022; Wang et al., 2022b). In reality, many entities share the same words and form nested entities that are part of another entity. About 53.9% entities are nested in the GENIA dataset (Kim et al., 2003).

Figure 1 (a) illustrates an instance from the GENIA dataset. A protein molecule entity "lipoxygenase" is nested within a protein family or group entity "lipoxygenase metabolites". Due to the overlapped part, nested entities are more likely to obtain similar representations from pre-trained language models, increasing the difficulty of distinguishing them. Besides, the few support instances could hardly represent the whole distribution, and prototype shifts often occur between the support and query sets in the few-shot scenario.

Prototype shift in NER refers to the change in prototypes between the support and query sets, as illustrated in Figure 1 (b), where the prototype is a representative example of a specific entity type and is usually calculated by averaging the representations of entities in the same class. Prototypes learned on the support set are typically used to identify the decision boundaries in the query set. Therefore, many classification errors would be introduced when the prototype shift happens. Besides, prototype shift is particularly harmful to nested entities due to their similar representations. Even though we have distinguished nested entities in the support set, they may still mix in the query set due to the prototype shift.

This paper addresses the prototype shift in the few-shot nested NER task. We regard entity representation as the combination of prototype and non-prototype representations. Entities of the same type should share the same prototype representation. And the non-prototype representation determines the dispersion of the entity distribution. If we could focus more on the prototype representation when learning the entity representation, entities would gather closer around the prototype, and the prototype shift could be reduced. Therefore, we design a prototype-attention mechanism to enhance the prototype representation. Besides, words of the O-type have miscellaneous semantics and cannot be represented by a unique prototype. Therefore, we further design a novel prototype-span contrastive loss. It compares prototype-enhanced span representations with original semantic representations to guarantee the O-type's representations are not enhanced by entity prototypes. It also compares prototype-enhanced span representations with prototypes to reduce prototype differences further.

Our main contributions are as follows:

- We identify the prototype shift challenge in the few-shot learning, particularly in the few-shot nested NER task, and propose a Prototype-Attention Contrastive Learning (PACL) framework to tackle it.
- We devise a novel Prototype-Attention mechanism to generate the prototype-enhanced representation for each span to mitigate the prototype shift between the support and query sets. This mechanism exhibits a high level of generality in enhancing the performance of two baseline models.
- We design a novel prototype-span contrastive loss by comparing prototype-enhanced span representations with prototypes and original semantic representations to reduce prototype differences further and overcome the O-type's non-unique prototype limitation.
- We conduct experiments on three nested NER datasets from three different languages. The results show improvements in PACL over existing nested NER and few-shot NER baselines in terms of F_1 score.

2 Problem Definition

Following the mainstream solutions, we formulate the few-shot nested NER task as a span-based entity classification problem. That is, given an input sentence $x \in \mathcal{X}$ with l tokens, denoted by $x = \{w_1, \ldots, w_l\}$, we generate an entity span set containing all possible spans, and each span s_{pq} is a span of tokens starting from the pth token and ending at the qth token in x, denoted by $s_{pq} = \{w_p, \ldots, w_q\}$ $(1 \leqslant p \leqslant q \leqslant l)$. Then, we learn a classification model to map each span into an entity label in the label set $E_{\mathcal{X}}$. If we set the task as a K-shot task, then the number of span labels for each entity type used for training is limited to K. Besides we also apply the meta-learning framework. The formal descriptions are as follows.

Let $\mathcal{D} = \{\mathcal{X}, \mathcal{Y}\}$ denote a dataset with $\mathcal{X}$ and $\mathcal{Y}$ as the sentence set and the corresponding label set, respectively. $\mathcal{D}^{spt} = \{\mathcal{X}^{spt}, \mathcal{Y}^{spt}\}$ and $\mathcal{D}^{qry} = \{\mathcal{X}^{qry}, \mathcal{Y}^{qry}\}$ are disjoint sets sampled from $\mathcal{D}$ for model training and testing, respectively. They are also known as the support set and the query set. Suppose $\mathcal{D}_i = \{\mathcal{X}_i, \mathcal{Y}_i\}$ and $\mathcal{D}_j = \{\mathcal{X}_j, \mathcal{Y}_j\}$ are the source and target domain datasets, respectively. The few-shot nested NER task first samples several subtasks $\{\mathcal{D}_i^{spt}, \mathcal{D}_i^{qry}\}$ from $\mathcal{D}_i = \{\mathcal{X}_i, \mathcal{Y}_i\}$, where $\mathcal{D}_i^{spt} = \{\mathcal{X}_i^{spt}, \mathcal{Y}_i^{spt}\}$, $\mathcal{D}_i^{qry} = \{\mathcal{X}_i^{qry}, \mathcal{Y}_i^{qry}\}$. It

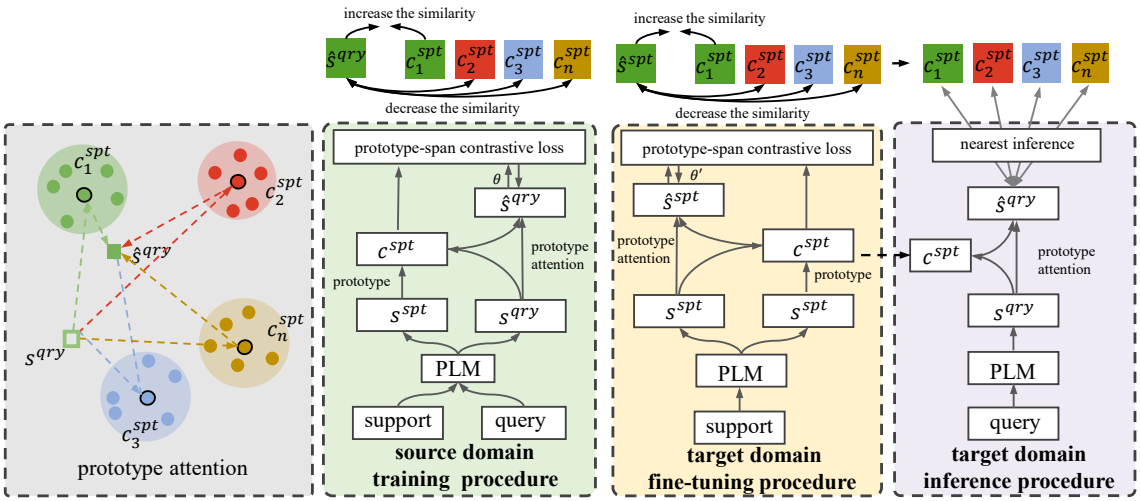


Figure 2: Illustration of our PACL framework and learning procedures. During the training procedure on the source domain, PACL calculates prototypes based on labeled spans of the support set and then utilizes prototype-attention to obtain prototype-enhanced representations for the query set. After that, PACL applies the prototype-span contrastive loss to optimize the representations. During the fine-tuning procedure on the target domain, PACL generates prototype-enhanced representations for the support set to fine-tune the model. Finally, PACL makes inferences on the query set of the target domain based on the nearest neighbor strategy.

then trains a model on these subtasks. After that, it makes adaptations on $\mathcal{D}_j$, i.e., it fine-tunes the model on $\mathcal{D}_j^{spt} = \{\mathcal{X}_j^{spt}, \mathcal{Y}_j^{spt}\}$ and then predicts the span labels for $\mathcal{D}_j^{qry} = \{\mathcal{X}_j^{qry}\}$. For the K-shot setting, each entity category in $\mathcal{X}_i^{spt}$ and $\mathcal{X}_j^{spt}$ contains K entities.

3 Methodology

This section introduces our PACL framework and then provides details of the prototype-attention mechanism, the prototype-span contrastive loss, and target domain adaption procedures.

3.1 PACL Framework

Figure 2 illustrates our Prototype-Attention Contrastive Learning (PACL) framework and learning procedures.

PACL first applies a Pre-trained Language Model (PLM) to obtain the semantic representation for each span. It then calculates prototypes on the support set and utilizes a novel prototype-attention mechanism to achieve prototype-enhanced representations. After that, PACL optimizes representations by a prototype-span contrastive loss.

During the training procedure on the source domain, PACL utilizes a bunch of subtasks $\{\mathcal{D}_i^{spt}, \mathcal{D}_i^{qry}\}$ to train the model. It generates prototype-enhanced representations for the query set to obtain the adjustment ability for prototype shift. During the fine-tuning procedure on the target domain, PACL utilizes $\mathcal{D}_j^{spt} = \{\mathcal{X}_j^{spt}, \mathcal{Y}_j^{spt}\}$ to fine-tune the model by generating prototype-enhanced representations for the support set. Finally, it predicts the labels for $\mathcal{D}_j^{qry} = \{\mathcal{X}_j^{qry}\}$ by the nearest neighbor strategy.

3.2 Prototype-Attention Mechanism

The intuitive idea for mitigating the prototype shift is to make the model focus more on prototype representations. Therefore, we design a prototype-attention mechanism to obtain prototypes on the support set and generate prototype-enhanced representations for the query set according to these prototypes when training on the source domain. In this way, span representations in the query set would contain more information about prototypes, which makes the prototypes of the query set be closer to those of the support set. The detailed procedures are as follows.

We first incorporate a Pre-trained Language Model (PLM) to obtain original span semantic representations. That is, for the sentence x with l tokens, we get all word embeddings, concatenate the start and the end token embeddings of each span, and use a non-linear function to get the span semantic representation $\boldsymbol{s}$:

$$[\boldsymbol{h_1}, \boldsymbol{h_2}, \ldots, \boldsymbol{h_l}] = \mathrm{PLM}([\mathrm{w}_1, \mathrm{w}_2, \ldots, \mathrm{w}_l]) \quad (1)$$

$$\boldsymbol{s} = f(\boldsymbol{h_p} \oplus \boldsymbol{h_q}) \quad (2)$$

Where $\oplus$ denotes the concatenation operator, and f is a non-linear function.

For the support set, we calculate the prototype $\boldsymbol{c}_k$ for each entity type k according to span labels except the O-type:

$$\boldsymbol{c}_k = \frac{1}{|\boldsymbol{s}_k|} \sum \boldsymbol{s}_k \tag{3}$$

Where $|\boldsymbol{s}_k|$ denotes the number of spans in type k.

For spans in the query set, we gain the prototype-enhanced representation $\hat{\boldsymbol{s}}^{qry}$ by calculating the attention score between the original span representation $\boldsymbol{s}^{qry}$ and prototypes $\mathcal{C} = [\boldsymbol{c}_1, \boldsymbol{c}_2, \ldots]$ in the support set:

$$\hat{\boldsymbol{s}}^{qry} = softmax(\frac{\boldsymbol{s}^{qry}\,\mathcal{C}^\top}{\sqrt{d_\mathcal{C}}})\,\mathcal{C} + \boldsymbol{s}^{qry} \tag{4}$$

where $d_\mathcal{C}$ is the dimension of prototypes. Note we also add $\boldsymbol{s}^{qry}$ to the attention representation to obtain $\hat{\boldsymbol{s}}^{qry}$. This is because the O-type could not be represented by prototypes in $\mathcal{C}$. Therefore, we want the O-type spans not to be enhanced by these known prototypes. This will be optimized by the following prototype-span contrastive loss in the next section.

3.3 Prototype-Span Contrastive Loss

The traditional contrastive loss increases span similarities of the same entity type and decreases span similarities between different entity types. This paper aims to address the prototype shift. Therefore, we want to increase the similarity between spans in the query set and the corresponding prototype in the support set to let the model obtain the ability to mitigate the prototype shift. Besides, the O-type span has miscellaneous semantics and could not be represented by a unique prototype (Fritzler et al., 2019). We also want prototype-enhanced representations of O-type entities close to their original semantic representations. Therefore, we design the following prototype-span contrastive loss based on the circle loss (Sun et al., 2020).

For each span representation $\hat{\boldsymbol{s}}^{qry}$ in the query set, the loss $\mathcal{L}_{\hat{\boldsymbol{s}}^{qry}}$ is calculated by:

$$\mathcal{L}_{\hat{\boldsymbol{s}}^{qry}} = log(1 + sim(\hat{\boldsymbol{s}}^{qry}, \boldsymbol{c}^+) * sim(\hat{\boldsymbol{s}}^{qry}, \boldsymbol{c}^-)) \tag{5}$$

Where $\boldsymbol{c}^+$ is the corresponding prototype in the support set with the same type as $\hat{\boldsymbol{s}}^{qry}$, and $\boldsymbol{c}^-$ denotes prototypes in the support set with different types from $\hat{\boldsymbol{s}}^{qry}$. The similarity function sim is calculated by:

$$sim(\hat{\boldsymbol{s}}^{qry}, \boldsymbol{c}^+) = e^{-\tau * \phi(\hat{\boldsymbol{s}}^{qry},\, \boldsymbol{c}^+)} \tag{6}$$

$$sim(\hat{\boldsymbol{s}}^{qry}, \boldsymbol{c}^-) = \sum_{\boldsymbol{c}_i^- \in \boldsymbol{c}^-} e^{\tau * \phi(\hat{\boldsymbol{s}}_i^{qry},\, \boldsymbol{c}_i^-)} \tag{7}$$

Where $\phi(.)$ denotes the cosine similarity, τ is the temperature (Wang and Liu, 2021).

When calculating $sim(\hat{\boldsymbol{s}}^{qry}, \boldsymbol{c}^+)$ for the O-type, we calculate the cosine similarity between the original span representation $\boldsymbol{s}^{qry}$ and the prototype-enhanced representation $\hat{\boldsymbol{s}}^{qry}$:

$$\phi(\hat{\boldsymbol{s}}_i^{qry},\ \boldsymbol{c}_o) = \lambda * \phi(\hat{\boldsymbol{s}}_i^{qry}, \boldsymbol{s}^{qry}) \tag{8}$$

Where λ is a learnable hyperparameter. We calculate the cosine similarity between the prototype-enhanced representation $\hat{\boldsymbol{s}}^{qry}$ and its corresponding prototype in the support set for other entity types.

3.4 Target Domain Adaption

After training the model on the source domain, we make adaptions to the target domain, including fine-tuning the model on the support set and making inferences on the query set.

During the fine-tuning procedure, our PACL first generates prototype-enhanced representations $\hat{\boldsymbol{s}}^{spt}$ for spans in the support set by calculating the attention score between the original span representation $\boldsymbol{s}^{spt}$ and the prototypes $\mathcal{C}$ in the support set. After that, PACL fine-tunes the model by utilizing the prototype-span contrastive loss with the input of $\hat{\boldsymbol{s}}^{spt}$ and $\mathcal{C}$. Different from using $\hat{\boldsymbol{s}}^{qry}$ as the input in the training procedure, we utilize $\hat{\boldsymbol{s}}^{spt}$ in the fine-tuning procedure since the labels of the query set are unknown. Using $\hat{\boldsymbol{s}}^{spt}$ is to make entities gather around prototypes, and this does not affect our PACL to mitigate the prototype shift. This is because the prototype shift is domain-independent. It is only related to the sampling strategy. Our PACL has already learned prototype shift patterns and acquired the ability to mitigate the prototype shift during training on the source domain.

During the inference procedure, our PACL obtains prototype-enhanced representations $\hat{\boldsymbol{s}}^{qry}$ for spans in the query set according to prototypes $\mathcal{C}$ in the support set. It further applies the nearest neighbor inference for each span according to the maximum similarity with prototypes or its original span representation (O-type).

Note that the support set in the target domain may be too large to load all spans in a single fine-tune procedure. For example, the support set of the GENIA dataset contains more than 20 sentences with over 5000 spans. Loading all sentences into a single batch to get prototypes for each type is challenging. Thus, for each sentence in the support set, we leave all entity spans and drop $1/|\mathcal{X}_j^{spt}|$ percentage of O-type spans during fine-tuning, where $|\mathcal{X}_j^{spt}|$ denotes the number of sentences in the support set.

4 Experiments

In this section, we evaluate the performance of the proposed PACL framework in the few-shot nested NER setting. After introducing the datasets and baseline models utilized, we outline the experimental setup, present experimental results, and provide a thorough analysis.

4.1 Datasets

To better assess the performance and generality of our proposed PACL framework across different languages, we chose the Indo-European language family for our experiments, as obtaining datasets in these languages is readily feasible. We use English as the source language and select three target languages based on their linguistic proximity to English. These include English, which is identical to the source language; German, a member of the Germanic branch of the Indo-European family to which English also belongs; and Russian, a member of the Slavic branch of the Indo-European family.

Dataset	language	Types	Sentences	Entities/Nest entities
GENIA	English	36	18.5k	55.7k / 30.0k
GermEval	German	12	18.4k	41.1k / 6.1k
NEREL	Russian	29	8.9k	56.1k / 18.7k
FewNERD	English	66	188.2k	491.7k / -

Table 1: Datasets used in experiments

As shown in Table 1, the target nested NER datasets are GENIA[1] in English (Kim et al., 2003), GermEval[2] in German (Benikova et al., 2014), and NEREL[3] in Russian (Loukachevitch et al., 2021). We use a flat NER dataset, FewNERD [4] in English (Ding et al., 2021), as the source domain dataset to train the model. All these datasets are publicly available under the licenses of CC-BY 3.0 for GENIA, CC-BY 4.0 for GermEval, CC-BY 2.5 for NEREL, and CC-BY-SA 4.0 for FewNERD. We have manually checked to guarantee these datasets are without offensive content and identifiers.

For training in the source domain, We randomly sampled 15,000 5-way 5-shot subtasks from the FewNERD inter-domain subset, among which 500 subtasks are used for validation. We validated the model every 1000 subtasks. When fine-tuning in the target domain, we sampled 32-way, 12-way, and 29-way support sets under 1-shot and 5-shot settings from GENIA, GermEval test subset, and NEREL test subset, respectively. We dropped four entity types in GENIA due to their number of entities being less than 50. After fine-tuning, we made inferences on the left instances in GENIA, GermEval test subset, and NEREL test subset.

4.2 Baselines

We compare our proposed PACL with seven baselines which can be categorized into three groups: 1) Rich-resource nested NER methods including NER-DP (Yu et al., 2020) and TIdentifier (Shen et al., 2021); 2) Metric-based few-shot NER methods including ProtoNet (Snell et al., 2017), NNShot (Yang and Katiyar, 2020), ESD (Wang et al., 2022c), and SpanProto (Wang et al., 2022a); 3) Contrastive-learning-based few-shot NER method CONTaiNER (Das et al., 2021). Appendix A details these baseline models. We train and fine-tune these baselines following the classical batch-based procedures.

4.3 Experimental Settings

We implemented PACL by Huggingface Transformer 4.21.1 and PyTorch 1.12.1. The model is initialized randomly and trained by AdamW (Loshchilov and Hutter, 2017). We train and fine-tune the model with the learning rate 5e-5. For the text encoder, we use the pre-trained $\text{BERT}_{\text{base_multilingual}}$ model with 768 hidden word-embedding dimensions for all datasets since the languages of target domain datasets are different. The hidden layer of the non-linear function f in equation 2 for getting span semantic representations is set to 512, and the initial value of the learnable hyperparameter λ for the O-type is set to 0.5. We set random seeds ranging from 0 to 10 to get ten results for each setting and report the average and standard deviation values to evaluate

[1] http://www.geniaproject.org/genia-corpus
[2] https://sites.google.com/site/germeval2014ner/data
[3] https://github.com/nerel-ds/NEREL
[4] https://ningding97.github.io/fewnerd/

Model/Framework	GENIA (32-way)		GermEval (12-way)		NEREL (29-way)		Average	
	1-shot	5-shot	1-shot	5-shot	1-shot	5-shot	1-shot	5-shot
NER-DP	15.26±2.78	31.89±4.01	7.12±2.61	24.89±3.92	15.86±5.77	42.25±2.42	12.75	33.01
TIdentifier	25.61±4.78	38.60±17.02	12.75 ±7.93	51.55±2.48	35.28±5.81	61.05±2.21	24.55	50.40
CONTaiNER	18.47±2.36	19.90±1.21	29.18±7.05	37.05±1.01	26.61±1.75	44.37±1.27	24.75	33.77
ProtoNet	19.76±1.73	38.01±2.75	33.20±9.00	47.95±4.06	38.70±4.62	50.22±1.28	30.55	45.39
NNShot	24.25±2.89	35.57±2.43	28.58±6.76	41.26±2.50	38.58±1.30	46.54±1.93	30.47	41.12
ESD	21.73±3.64	27.54±3.17	34.00±8.75	34.75±6.03	28.56±5.18	47.68±2.20	28.10	36.66
SpanProto	29.01±3.55	41.84±2.66	34.12±6.64	51.11±5.89	44.20±3.55	56.16±2.15	35.78	49.70
PACL	**37.92**±1.97	**49.58**±1.82	**53.51**±7.70	**65.87**±1.80	**51.76**±2.85	**65.12** ±1.97	**47.73**	**60.19**

Table 2: F_1 performance on GENIA, GermEval, and NEREL datasets with 1-shot and 5-shot settings (%).

all models. We run all the experiments on a single NVIDIA A10 GPU.

4.4 Experimental Results

To evaluate the effectiveness of our PACL, we compare it against state-of-the-art baseline models introduced in 4.2. Table 2 shows their average F_1 results on GENIA, GermEval, and NEREL nested NER datasets with 1-shot and 5-shot settings. The proposed PACL outperforms these models.

On the GENIA dataset, our PACL outperforms the other models on both 1-shot and 5-shot settings, with F_1 scores of 37.92% and 49.58% respectively, compared to the best-performing baseline model, SpanProto, which scores 29.01% and 41.84%, respectively.

Our PACL also demonstrates superior performance on the GermEval dataset, scoring 53.51% on 1-shot and 65.87% on 5-shot, compared to the best-performing baseline model on 1-shot SpanProto with 34.12% and the best-performing baseline model on 5-shot TIdentifier with 51.55%.

Finally, on the NEREL dataset, our PACL again outperforms the other models, scoring 51.76% on 1-shot and 65.12% on 5-shot, compared to the best-performing baseline model on 1-shot SpanProto with 44.20% and the best-performing baseline model on 5-shot TIdentifier with 61.05%.

Overall, these results demonstrate the effectiveness of our proposed PACL framework compared to the state-of-the-art baseline models.

4.5 Experimental Analysis

This section presents ablation studies, prototype shifts across various shots, and the generality of the prototype-attention mechanism.

4.5.1 Ablation Study

To evaluate the contribution of the designed Prototype-Attention (PA) mechanism to the overall performance of PACL, we conduct the ablation study by removing PA from the PACL. The results in Table 3 suggest that the PA mechanism positively impacts the F_1 score for the GENIA, GermEval, and NEREL datasets. To be specific, the PA mechanism improves the F_1 score by 3.21%, 3.77%, and 3.66% on the GENIA, GermEval, and NEREL datasets with the 1-shot setting, respectively. It also leads to 1.28%, 3.58%, and 0.87% increases on the GENIA, GermEval, and NEREL datasets with the 5-shot setting, respectively.

Overall, the results of this ablation study demonstrate that the PA mechanism enhances performance on various datasets with a particularly pronounced impact in 1-shot settings. The noticeable difference between the performance improvements in the 1-shot and 5-shot settings provokes our interest in understanding how prototype shifts behavior across different shot settings, which will be the focus of the next section.

Note we did not explore the influence of the prototype-span contrastive loss by replacing it with a classical contrastive loss. This is because the classical contrastive loss performs similarity measurement between span and span. In PACL, the span representations for the query set are enhanced by the PA mechanism while the span representations for the support set are not enhanced. Comparing these two different types of span representations is

		PACL	w/o PA
GENIA	1-shot	37.92±1.97	34.71±1.82
	5-shot	49.58±1.82	48.30±2.37
GermEval	1-shot	53.51±7.70	49.74±7.61
	5-shot	65.87±1.80	62.29±2.86
NEREL	1-shot	51.76±2.85	48.10±3.69
	5-shot	65.12±1.97	64.25±1.21

Table 3: Ablation study of F_1 performance on three datasets (%). "**w/o PA**" means removing the Prototype-Attention mechanism.

inappropriate. Therefore, we just study the effectiveness of the PA mechanism.

4.5.2 Prototype Shift Study

	GENIA(32-way)				
	1-shot	3-shot	5-shot	7-shot	9-shot
dis	8.61	5.71	4.85	4.36	3.84
	GermEval(12-way)				
	1-shot	3-shot	5-shot	7-shot	9-shot
dis	9.92	6.18	5.03	4.66	4.05
	NEREL(29-way)				
	1-shot	3-shot	5-shot	7-shot	9-shot
dis	8.06	5.29	4.30	3.73	3.29

Table 4: Prototype shifts across various shots on three nested datasets. "**dis**" means the euclidean distance between prototypes in the support set and those in the query set.

This section examines how prototype shifts behavior across different shot settings on the given three datasets. As discussed earlier, prototype shifts occur when the prototypes learned from the support set differ from those in the query set, which may affect performance. We use the Euclidean distance between prototypes to measure the prototype shift.

The findings in Table 4 reveal a consistent pattern of increasing Euclidean distance between prototypes as the number of shots decreases. While this pattern conforms to the sampling theory, it highlights that prototype shifts are more significant for smaller shot sizes where the proposed PACL is expected to be most effective. Appendix B shows how our PACL mitigates the prototype shift.

4.5.3 Generality of Prototype-Attention Mechanism

As the Prototype-Attention (PA) mechanism addresses the fundamental property of the prototype shift phenomenon, we believe it has a high level of generality and can enhance the performance of various models.

		SpanProto	SpanProto w PA
GENIA	1-shot	29.01±1.97	33.17±4.53
	5-shot	41.84±2.66	44.76±2.00
GermEval	1-shot	34.12±6.64	44.03±9.26
	5-shot	51.11±5.89	54.90±1.99
NEREL	1-shot	44.20±3.55	49.04±3.03
	5-shot	56.16±2.15	64.17±1.5

Table 5: F_1 performance before and after integrating the Prototype-Attention (PA) mechanism to SpanProto on three datasets (%).

To assess the generality of the PA mechanism, we conduct experiments by integrating it into the

		ESD	ESD w PA
GENIA	1-shot	21.73±3.64	25.10±4.43
	5-shot	27.54±3.17	36.03±3.92
GermEval	1-shot	34.00±8.75	36.08±6.89
	5-shot	34.75±6.03	41.95±7.53
NEREL	1-shot	28.56±5.18	41.38±4.93
	5-shot	47.68±2.20	56.03±2.47

Table 6: F_1 performance before and after integrating the Prototype-Attention (PA) mechanism to ESD on three datasets (%).

SpanProto and ESD models and comparing the performance before and after integration. As shown in Table 5 and Table 6, the experiment results demonstrate that integrating the PA mechanism into SpanProto and ESD notably improves the F1 score on GENIA, GermEval, and NEREL datasets in both 1-shot and 5-shot settings.

These findings suggest that the PA mechanism has high generality and can serve as a valuable tool for NLP practitioners looking to improve their models' performance in few-shot nested NER tasks.

5 Related Work

This section discusses related works on rich-resource nested NER, few-shot NER, and distribution shifts.

5.1 Rich-resource Nested NER

Nested NER aims to recognize entities with nested structures. Most of the current methods for nested NER are established on rich-resource datasets, and they require a large number of instances for training the model. These methods could be categorized into span-based, hypergraph-based, and layered-based (Wan et al., 2022).

Span-based methods treat sequences of tokens as spans and then label all possible spans by classification models (Shen et al., 2021; Li et al., 2020b; Tan et al., 2021). Hypergraph-based methods analyze the dependence of words in a sentence and then construct a dependency tree (Yu et al., 2020) or other structures (Wang and Lu, 2018; Katiyar and Cardie, 2018) to help identify nested entities. And layered-based methods capture the depth of entity nesting and apply multi-level sequence labeling strategies to recognize nested entities (Wang et al., 2021; Shibuya and Hovy, 2020).

These methods may be stuck in overfitting due to sophisticated models and the limited number of instances for training in the few-shot setting.

5.2 Few-shot NER

Few-shot NER requires recognizing entities with the support of only very few labeled instances (Hofer et al., 2018; Fritzler et al., 2019). Due to limited information contained in support instances, methods for few-shot NER mainly resort to a rich-resource source domain to help train models, resulting in transfer-learning and meta-learning frameworks.

Transfer-learning-based methods train models on a source domain and then transfer models or features to the few-labeled target domain (Yang et al., 2021; Liu et al., 2021). Meta-learning-based methods train models on adequate subtasks to make the model acquire the learning ability on few-shot tasks (de Lichy et al., 2021; Li et al., 2020a). Comparatively speaking, meta-learning-based methods are more widely used in few-shot NER due to their easy adaption to new tasks.

Within the meta-learning framework, various kinds of models are designed. For example, metric-based methods, including ProtoNet (Snell et al., 2017), NNShot (Yang and Katiyar, 2020), and SpanProto (Wang et al., 2022a), measure distances between prototypes in the support set and instances in the query set. Optimization-based methods, such as MAML (Finn et al., 2017) and FEWNER (Li et al., 2020a), train the model by a special optimizer. Model-based methods, such as SNAIL (Mishra et al., 2017) and CNPs (Garnelo et al., 2018), learn the hidden representation of instances on the support set and the query set to make inferences in an end-to-end manner. Contrastive-learning methods, such as CONTaiNER (Das et al., 2022), aims to maximize similarities of the same type and minimize similarities between different types.

These few-shot NER methods mostly focus on flat entities. Few works have discussed the few-shot nested NER setting. Wang et al. converted sequence labeling to span-level matching for the few-shot flat NER and showed their method could handle nested entities (Wang et al., 2022b). However, it is not designed for the few-shot nested NER specifically.

5.3 Distribution Shifts

Distribution shift is a problem of training and testing data following two different distributions. It affects the generalization ability of supervised deep-learning models as the fundamental that these models could work is that training and testing data come from the same distribution. Inspired by real-world challenges, Wiles et al. summarized three distribution shifts: spurious correlation, low-data drift, and unseen data shift (Wiles et al., 2022). There have been some researches aiming to address distribution shifts in computer vision and general natural language processing tasks (Fang et al., 2020; Tu et al., 2022). To the best of our knowledge, researchers seldom discuss the distribution shift problem in the few-shot NER task. In this paper, we aim to tackle the few-shot nested NER task. Therefore, we rethink the distribution shift problem from the perspective of entity representation distribution and identify the prototype shift since it directly affects entity classification.

6 Conclusion

This paper first identifies the phenomenon of prototype shift that arises when there is a difference in prototypes between the support and query sets. Within the context of few-shot learning tasks, prototype shift is prone to occur since the few labeled instances in the support set could hardly represent the query set. To mitigate this issue in the few-shot nested NER task, we propose the Prototype-Attention Contrastive Learning (PACL) framework combining a prototype-attention mechanism and a prototype-span contrastive loss to enhance prototype representations. The experiments on three English, German, and Russian nested NER datasets demonstrated that PACL outperformed baseline models on the 1-shot and 5-shot settings. Future studies could explore the generality of PACL to other few-shot learning tasks.

7 Limitations

This paper still has several limitations. The first one is about the prototype shift adjustment. It is hard to completely address the prototype shift, while our PACL makes this attempt and achieves inspiring improvement. The second one is about other distribution shifts. Prototype shift is just one kind of distribution shift. Other distribution shifts also need to be identified and addressed to improve the accuracy of the few-shot nested NER task. The third one is about the language used for training. The results validate the performance of PACL across different languages, while the three languages used in this paper belong to the Indo-European family. This may introduce language bias to this language family and cause potential risk.

References

Darina Benikova, Chris Biemann, and Marc Reznicek. 2014. Nosta-d named entity annotation for german: Guidelines and dataset. In *LREC*, pages 2524–2531.

Sarkar Snigdha Sarathi Das, Arzoo Katiyar, Rebecca Passonneau, and Rui Zhang. 2022. CONTaiNER: Few-shot named entity recognition via contrastive learning. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 6338–6353, Dublin, Ireland. Association for Computational Linguistics.

Sarkar Snigdha Sarathi Das, Arzoo Katiyar, Rebecca J Passonneau, and Rui Zhang. 2021. Container: Few-shot named entity recognition via contrastive learning. *arXiv preprint arXiv:2109.07589*.

Cyprien de Lichy, Hadrien Glaude, and William Campbell. 2021. Meta-learning for few-shot named entity recognition. In *Proceedings of the 1st Workshop on Meta Learning and Its Applications to Natural Language Processing*, pages 44–58.

Ning Ding, Guangwei Xu, Yulin Chen, Xiaobin Wang, Xu Han, Pengjun Xie, Hai-Tao Zheng, and Zhiyuan Liu. 2021. Few-nerd:a few-shot named entity recognition dataset. In *ACL-IJCNLP*.

Tongtong Fang, Nan Lu, Gang Niu, and Masashi Sugiyama. 2020. Rethinking importance weighting for deep learning under distribution shift. *Advances in Neural Information Processing Systems*, 33:11996–12007.

Chelsea Finn, Pieter Abbeel, and Sergey Levine. 2017. Model-agnostic meta-learning for fast adaptation of deep networks. In *International conference on machine learning*, pages 1126–1135. PMLR.

Omar U Florez and Erik Mueller. 2019. Learning to control latent representations for few-shot learning of named entities. *arXiv preprint arXiv:1911.08542*.

Alexander Fritzler, Varvara Logacheva, and Maksim Kretov. 2019. Few-shot classification in named entity recognition task. In *Proceedings of the 34th ACM/SIGAPP Symposium on Applied Computing*, pages 993–1000.

Marta Garnelo, Dan Rosenbaum, Christopher Maddison, Tiago Ramalho, David Saxton, Murray Shanahan, Yee Whye Teh, Danilo Rezende, and SM Ali Eslami. 2018. Conditional neural processes. In *International Conference on Machine Learning*, pages 1704–1713. PMLR.

Maximilian Hofer, Andrey Kormilitzin, Paul Goldberg, and Alejo Nevado-Holgado. 2018. Few-shot learning for named entity recognition in medical text. *arXiv preprint arXiv:1811.05468*.

Yutai Hou, Cheng Chen, Xianzhen Luo, Bohan Li, and Wanxiang Che. 2022. Inverse is better! fast and accurate prompt for few-shot slot tagging. In *Findings of the Association for Computational Linguistics: ACL 2022*, pages 637–647, Dublin, Ireland. Association for Computational Linguistics.

Yutai Hou, Zhihan Zhou, Yijia Liu, Ning Wang, Wanxiang Che, Han Liu, and Ting Liu. 2019. Few-shot sequence labeling with label dependency transfer and pair-wise embedding. *arXiv preprint arXiv:1906.08711*.

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.

J-D Kim, Tomoko Ohta, Yuka Tateisi, and Jun'ichi Tsujii. 2003. Genia corpus—a semantically annotated corpus for bio-textmining. *Bioinformatics*, 19(suppl_1):i180–i182.

Jing Li, Billy Chiu, Shanshan Feng, and Hao Wang. 2020a. Few-shot named entity recognition via meta-learning. *IEEE Transactions on Knowledge and Data Engineering*.

Xiaoya Li, Jingrong Feng, Yuxian Meng, Qinghong Han, Fei Wu, and Jiwei Li. 2020b. A unified MRC framework for named entity recognition. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 5849–5859, Online. Association for Computational Linguistics.

Pengfei Liu, Weizhe Yuan, Jinlan Fu, Zhengbao Jiang, Hiroaki Hayashi, and Graham Neubig. 2021. Pre-train, prompt, and predict: A systematic survey of prompting methods in natural language processing. *arXiv preprint arXiv:2107.13586*.

Ilya Loshchilov and Frank Hutter. 2017. Fixing weight decay regularization in adam. *CoRR*, abs/1711.05101.

Natalia Loukachevitch, Ekaterina Artemova, Tatiana Batura, Pavel Braslavski, Ilia Denisov, Vladimir Ivanov, Suresh Manandhar, Alexander Pugachev, and Elena Tutubalina. 2021. NEREL: A Russian dataset with nested named entities, relations and events. In *Proceedings of RANLP*, pages 876–885.

Ruotian Ma, Xin Zhou, Tao Gui, Yiding Tan, Linyang Li, Qi Zhang, and Xuanjing Huang. 2022. Template-free prompt tuning for few-shot NER. In *Proceedings of the 2022 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 5721–5732, Seattle, United States. Association for Computational Linguistics.

Hong Ming, Jiaoyun Yang, Lili Jiang, Yan Pan, and Ning An. 2022. Few-shot nested named entity recognition. *arXiv preprint arXiv:2212.00953*.

Nikhil Mishra, Mostafa Rohaninejad, Xi Chen, and Pieter Abbeel. 2017. A simple neural attentive meta-learner. *arXiv preprint arXiv:1707.03141*.

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 *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 2782–2794, Online. 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.

Jake Snell, Kevin Swersky, and Richard S Zemel. 2017. Prototypical networks for few-shot learning. *arXiv preprint arXiv:1703.05175*.

Yifan Sun, Changmao Cheng, Yuhan Zhang, Chi Zhang, Liang Zheng, Zhongdao Wang, and Yichen Wei. 2020. Circle loss: A unified perspective of pair similarity optimization. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 6398–6407.

Flood Sung, Yongxin Yang, Li Zhang, Tao Xiang, Philip HS Torr, and Timothy M Hospedales. 2018. Learning to compare: Relation network for few-shot learning. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 1199–1208.

Zeqi Tan, Yongliang Shen, Shuai Zhang, Weiming Lu, and Yueting Zhuang. 2021. 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.

Jianhong Tu, Ju Fan, Nan Tang, Peng Wang, Chengliang Chai, Guoliang Li, Ruixue Fan, and Xiaoyong Du. 2022. Domain adaptation for deep entity resolution. In *SIGMOD '22: International Conference on Management of Data, Philadelphia, PA, USA, June 12 - 17, 2022*, pages 443–457. ACM.

Juncheng Wan, Dongyu Ru, Weinan Zhang, and Yong Yu. 2022. Nested named entity recognition with span-level graphs. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 892–903, Dublin, Ireland. Association for Computational Linguistics.

Bailin Wang and Wei Lu. 2018. Neural segmental hypergraphs for overlapping mention recognition. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 204–214, Brussels, Belgium. Association for Computational Linguistics.

Feng Wang and Huaping Liu. 2021. Understanding the behaviour of contrastive loss. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 2495–2504.

Jianing Wang, Chengyu Wang, Chuanqi Tan, Minghui Qiu, Songfang Huang, Jun Huang, and Ming Gao. 2022a. Spanproto: A two-stage span-based prototypical network for few-shot named entity recognition. *CoRR*, abs/2210.09049.

Peiyi Wang, Runxin Xu, Tianyu Liu, Qingyu Zhou, Yunbo Cao, Baobao Chang, and Zhifang Sui. 2022b. An enhanced span-based decomposition method for few-shot sequence labeling. In *Proceedings of the 2022 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 5012–5024, Seattle, United States. Association for Computational Linguistics.

Peiyi Wang, Runxin Xu, Tianyu Liu, Qingyu Zhou, Yunbo Cao, Baobao Chang, and Zhifang Sui. 2022c. An enhanced span-based decomposition method for few-shot sequence labeling. In *Proceedings of the 2022 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 5012–5024, Seattle, United States. 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 *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 3547–3557, Online. Association for Computational Linguistics.

Olivia Wiles, Sven Gowal, Florian Stimberg, Sylvestre-Alvise Rebuffi, Ira Ktena, Krishnamurthy Dj Dvijotham, and Ali Taylan Cemgil. 2022. A fine-grained analysis on distribution shift. In *International Conference on Learning Representations*.

Guanqun Yang, Shay Dineen, Zhipeng Lin, and Xueqing Liu. 2021. Few-sample named entity recognition for security vulnerability reports by fine-tuning pre-trained language models. In *International Workshop on Deployable Machine Learning for Security Defense*, pages 55–78. Springer.

Yi Yang and Arzoo Katiyar. 2020. Simple and effective few-shot named entity recognition with structured nearest neighbor learning. *arXiv preprint arXiv:2010.02405*.

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

A Detail of Baselines

Detailed information on baseline models is introduced in this section. We compare our PACL with the following seven baseline models:

- NER-DP (Yu et al., 2020) is a rich-resource-based nested NER method. It applies a biaffine model to score pairs of start and end tokens for each span to establish dependency parsing for identifying nested entities.
- TIdentifier (Shen et al., 2021) is also a rich-resource-based nested NER method. It applies a Two-stage Identifier (TIdentifier), including a seed span generation module for locating entities and a span proposal module for classifying entities.
- CONTaiNER (Das et al., 2021) is a contrastive-learning-based few-shot NER method. It first obtains entities' Gaussian-distributed embeddings and then optimizes a generalized objective of differentiating between entity types by a contrastive loss function. We adapt it to handle nested entities with the entity span formulation.
- ProtoNet (Snell et al., 2017) is a metric-learning-based few-shot NER method. It applies prototypical networks to learn a metric space for obtaining prototype representations. We also adapt it to handle nested entities with the entity span formulation.
- NNShot (Yang and Katiyar, 2020) is also a metric-learning-based few-shot NER method. It applies structured decoding and nearest-neighbor learning to identify entities. We utilize the entity span formulation to make it handle nested entities.
- ESD (Wang et al., 2022c) is a metric-learning-based few-shot NER method. It formulates the task as a span-level matching problem. To identify entities, it performs span-level procedures, including enhanced span representation, class prototype aggregation, and span conflict resolution.
- SpanProto (Wang et al., 2022a) is a metric-learning-based few-shot NER method. It also applies entity spans to formulate the problem. For identifying entities, it first utilizes a span extractor to recognize candidate entity spans and then applies a mention classifier to determine entity types.

B Prototype Shift Mitigation by PACL

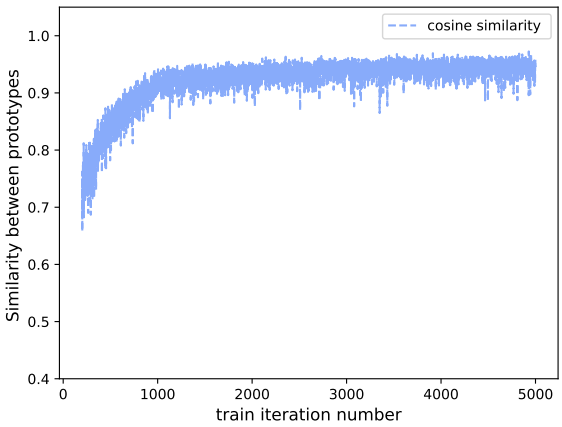


Figure 3: Illustration of the change of the prototype similarity during training.

This paper aims to mitigate prototype shifts, and section 4.5.2 has already validated the existence of the prototype shift phenomenon. This section examines how the prototype shift changes by applying our PACL.

We utilize the cosine similarity to denote the prototype differences between the support and query sets to measure the prototype shift. Figure 3 illustrates the change of the prototype similarity with the increase of iteration numbers during training. We could find a consistently increasing trend in prototype similarity, which means the prototype shift is consistently decreasing. This validates the effectiveness of our PACL in mitigating prototype shifts.

Note we do not apply the euclidean distance to measure the prototype shift because the embedding space keeps changing during training. For two spaces with different dimensions and scales, the euclidean distance could not faithfully reflect the differences between prototypes. On the other hand, using the euclidean distance is a better choice in the same space, therefore we apply the euclidean distance in section 4.5.2 to measure prototype differences in the space generated by BERT.