COPNER: Contrastive Learning with Prompt Guiding for Few-shot Named Entity Recognition

Yucheng Huang[1] **Kai He**[1] **Yige Wang**[1] **Xianli Zhang**[1]
Tieliang Gong[1] **Rui Mao**[2] **Chen Li**[1*]
[1]School of Computer Science and Technology, Xi'an Jiaotong University
[2]School of Computer Science and Engineering, Nanyang Technological University
{huangyucheng, hk52025804, jihejue039, xlbryant}@stu.xjtu.edu.cn
{gongtl, cli}@xjtu.edu.cn, rui.mao@ntu.edu.sg

Abstract

Distance metric learning has become a popular solution for few-shot Named Entity Recognition (NER). The typical setup aims to learn a similarity metric for measuring the semantic similarity between test samples and referents, where each referent represents an entity class. The effect of this setup may, however, be compromised for two reasons. First, there is typically a limited optimization exerted on the representations of entity tokens after initing by pre-trained language models. Second, the referents may be far from representing corresponding entity classes due to the label scarcity in the few-shot setting. To address these challenges, we propose a novel approach named COntrastive learning with Prompt guiding for few-shot NER (COPNER). We introduce a novel prompt composed of class-specific words to COPNER to serve as 1) supervision signals for conducting contrastive learning to optimize token representations; 2) metric referents for distance-metric inference on test samples. Experimental results demonstrate that COPNER outperforms state-of-the-art models with a significant margin in most cases. Moreover, COPNER shows great potential in the zero-shot setting. The source code is available at: https://github.com/AndrewHYC/COPNER.

1 Introduction

As a fundamental task in Nature Language Process (NLP), Named Entity Recognition (NER) aims to identify the spans of text according to a pre-defined set of entity classes, such as person, organization and location (Sang and De Meulder, 2003). Many down-stream tasks heavily rely on these extracted entities, such as aspect-level sentiment (Mao and Li, 2021), intention recognition (Vedula et al., 2020), and knowledge graph construction (He et al., 2021). An enormous number of neural methods have shown promising ability on NER tasks (Chiu and Nichols, 2016; Yadav and Bethard, 2018; Li et al., 2020), whereas insufficient labeled data in different domains are still a significant challenging for the community. Considering obtaining full annotated data is labor-intensive and time-consuming, few-shot NER studies (Fritzler et al., 2019; Yang and Katiyar, 2020; Das et al., 2021; Cui et al., 2021; Ma et al., 2021) are raising more attention, which can alleviate annotation dependence and help neural methods transfer to other tasks easier.

Recently, Prompt Learning (PL) becomes a popular technology in NLP and shows great potential for dealing with few-shot issues (Liu et al., 2021b; Ding et al., 2021b; Chen et al., 2021; Mao et al., 2022). Typical prompt learning is designed for understanding sentence-level tasks by decoding a special marker of an input. However, it is challenging to adapt PL to token-level tasks, which needs to identify the class of each token. Inspired by PL, TemplateNER (Cui et al., 2021) applies manual templates for few-shot NER, which needs to enumerate all potential spans and forward propagate many times for each input, which is time-consuming.

Compared with TemplateNER, distance metirc-based approaches (Wiseman and Stratos, 2019; Yang and Katiyar, 2020; Ziyadi et al., 2020) are more popular and efficient in few-shot NER tasks. The key idea of this paradigm is to learn a similarity metric for measuring the semantic similarity between test samples and referents (e.g., prototypes or the nearest neighbors). The referents are usually derived from a few labeled samples through a pre-trained language model, e.g., BERT (Devlin et al., 2019). While being less costly, these methods are still limited in two aspects: Firstly, their ability to capture entity-class-related semantics is limited, because their main goal is to learn a suitable similarity function rather than optimizing the parameters of a sentence encoder for better entity representations. Secondly, as an anchored metric

[*]Corresponding author

referent is derived from only few labeled data, it is insufficient to properly represent the semantics of the corresponding entity class. This issue may severely impact the performance of a few-shot NER model in the inference phase.

To address these issues, we propose a novel approach named **CO**ntrastive learning with **P**rompt guiding for few-shot **NER** (COPNER). The core idea of COPNER is to leverage class-specific words (CWs) from natural language to serve as the agents of corresponding entity types. Specifically, the CWs are included by appending a prompt to the original input sentence. As shown in Figure 1(a), an original input sentence "`[BOS] Obama was born in 1961 [EOS]`" is concatenated with a prompt "`person date none`", where "`person`", "`date`" and "`none`" are the CWs for the entity classes of person, date, and non-entity, respectively. In the training phase, the representations of CWs are served as token-level supervision signals that guide the sentence encoder to pull the representations of tokens belonging to the same class, and also the representation of the anchored CW, to be closer. In this way, the sentence encoder can learn to capture the dependence between tokens for aligning the semantics of a mentioned entity with the semantics of the corresponding CW in the unified semantic space. In the inference phase, the representations of CWs are treated as metric referents for predicting entity classes. As the representations of CWs contain general and discriminates semantics at the scratch and are further trained to align with the corresponding entity set, they are more appropriate and stable than the referents derived from previous works. Further, we explore different methods of prompt construction, aiming to understand the effects of different forms of prompts.

We summarize existing few-shot NER research into three settings, including Cross-Label-Space, Domain Transfer, and In-Label-Space. COPNER is evaluated under all these settings. We conduct experiments on six NER datasets and COPNER largely outperforms state-of-the-art(SOTA) approaches in most cases, especially in complex scenarios. Specifically, compared with SOTA results in Few-NERD (INTRA) and Few-NERD (INTER), COPNER raises the F1 scores of 8.28% and 8.03% in these two fine-grained few-shot NER tasks. In Domain Transfer settings, COPNER also improves by 9.0% averaged F1 scores under 1-shot settings. Additionally, considering CWs can inherently carry the relevant category information, we explore the zero-shot ability of COPNER with satisfied results. The main contributions of this paper are summarized as follows:

- We propose a novel few-shot NER approach named COPNER, which combines contrastive learning and prompt guiding. By introducing prompts as supervision signals and metric referents, COPNER overcomes the problem that typical class referents cannot properly represent each category due to data scarcity, and enhances entity representations for better discrimination with contrastive learning.
- We detailly investigate the existing few-shot NER research and summarize them into three categories. COPNER is evaluated in all these categories and outperforms SOTAs in most cases. Further, we expand the boundaries of COPNER's ability to the zero-shot setting and also achieve satisfied results.

2 Task Formulation

NER is normally treated as a sequence labeling task. For each input sentence $X = \{x_1, x_2, ..., x_t\}$, NER models aim to assign each token x_i a label $y_i \in C$, where C is a predefined label set. The assigned label shows either the token is a part of a named entity or out of any entity classes.

For each entity class, few-shot NER tasks are only provided with very limited annotations as supervision for neural models. With a comprehensive survey, we summarize existing few-shot NER research into three settings, and COPNER is evaluated in all these settings to demonstrate its effectiveness and universality.

2.1 Cross-Label-Space Setting

For few-shot NER with Cross-Label-Space setting, there is a rich-resource dataset (source set $\mathbb{H}$) for training and a low-resource dataset (target set $\mathbb{L}$) for adapting and testing. Notably, the target label space $\mathcal{C}^L$ is totally different from the source label space $\mathcal{C}^H$, namely $\mathcal{C}^L \cap \mathcal{C}^H = \emptyset$. Usually, the Cross-Label-Space setting is combined with N-way K-shot setting, e.g., a support set consists of N entity classes and each class has K labeled examples. To deal with this setting, an NER model needs to be trained on $\mathbb{H}$ first, then adopts to a new label space using the support set of $\mathbb{L}$. It is challenging

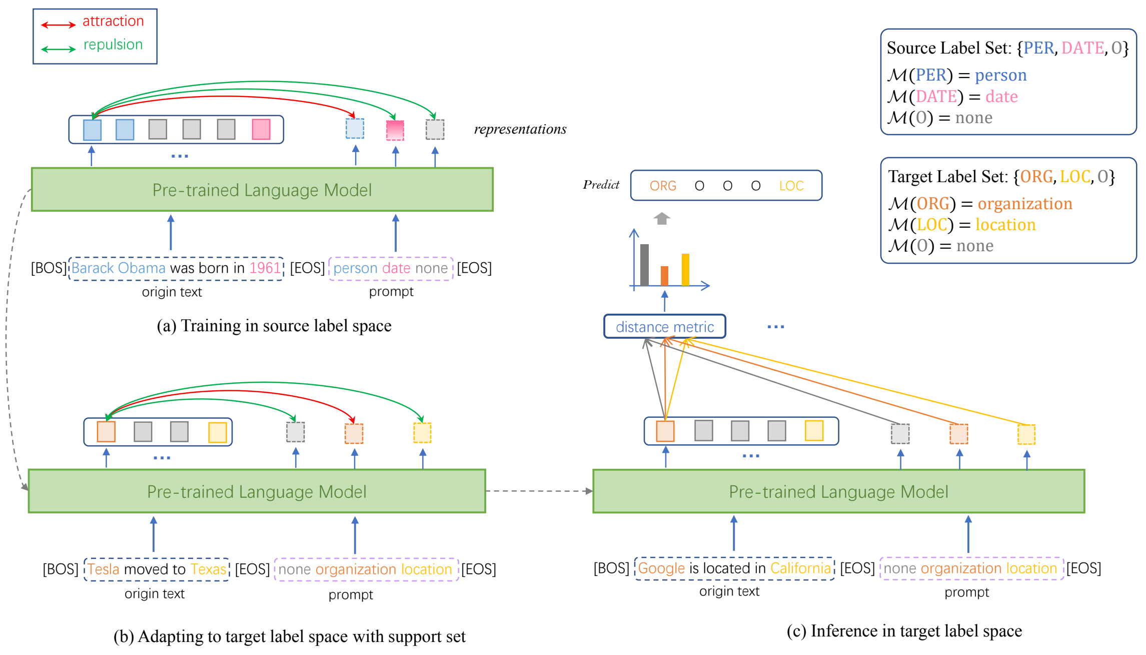

attraction
repulsion
representations
Pre-trained Language Model
[BOS] Barack Obama was born in 1961 [EOS] person date none [EOS]
origin text
prompt
(a) Training in source label space
Predict
ORG O O O LOC
distance metric
Source Label Set: {PER, DATE, O}
$\mathcal{M}$(PER) = person
$\mathcal{M}$(DATE) = date
$\mathcal{M}$(O) = none
Target Label Set: {ORG, LOC, O}
$\mathcal{M}$(ORG) = organization
$\mathcal{M}$(LOC) = location
$\mathcal{M}$(O) = none
Pre-trained Language Model
[BOS] Tesla moved to Texas [EOS] none organization location [EOS]
origin text
prompt
(b) Adapting to target label space with support set
Pre-trained Language Model
[BOS] Google is located in California [EOS] none organization location [EOS]
origin text
prompt
(c) Inference in target label space

Figure 1: The illustration of the COPNER framework based on Contrastive Learning with Prompt guiding: (a) Training in the source label space {PER, DATE, O}. (b) Adapting to the target label space {ORG, LOC, O} with support set. (c) Inferring by comparing test tokens with class-specific words.

that COPNER needs to handle the problems of low resources and cross-label space.

2.2 Domain Transfer Setting

Similar to the Cross-Label-Space setting, there is also a rich-resource dataset (source set $\mathbb{H}$) for training and a low-resource dataset (target set $\mathbb{L}$) for adapting. The main difference is these two datasets come from different domains. For example, $\mathbb{H}$ can be news corpus while $\mathbb{L}$ comes from medical data. Besides, the label spaces of $\mathbb{H}$ and $\mathbb{L}$ can overlap. This setting needs COPNER to keep the domain transfer capability with limited annotations.

2.3 In-Label-Space Setting

Different from the previous two settings that have a rich-resource dataset, the In-Label-Space setting supposes that only a small number of labeled examples can be used for training. Specifically, a few-shot NER model is first trained on a dataset $\mathbb{D}^{train}$ with a label space $\mathcal{C}$, in which each entity class has only K samples. Then, the model is evaluated by a test set $\mathbb{D}^{test}$ with the same label space $\mathcal{C}$. It is a great challenge that COPNER needs to learn the NER task with only few training samples.

3 Methodology

The key idea of COPNER is to construct prompts with CWs for both the model training and inference. For each sentence, the COPNER concatenates a prompt to it and feeds the supplemented one into a pre-trained language model (PLM). Then, the PLM is trained in a contrastive learning fashion. In this process, the prompts play the role of the token-level anchors for constructing the positive pairs and negative pairs of contrastive learning. Finally, in the inference stage, the representations of CWs (in the prompt) are served as metric referents for predicting.

The whole process of COPNER is shown in Figure 1. We first train the used PLM in source label spaces. Next, the PLM is fine-tuned by a few support sets for adapting to new label spaces in evaluation tasks, e.g., Cross-Label-Space and Domain Transfer. Finally, a token-level distance-based metric classifier is employed to obtain the final results in the inference phase.

3.1 Prompt Guided Few-shot NER Model

First, our method constructs task-specific prompts for an employed dataset. A class-specific word mapping $\mathcal{M}$ is manually defined, then a unique class-specific word v_i is obtained for each entity label $c_i \in C$ with $\mathcal{M}$, where C is the predefined label set. We use a simple yet effective method to develop $\mathcal{M}$, i.g. using class names of the given entities as CWs. For example, "`LOC`" is used as the label for location entities in most NER datasets, then the class-specific word "`location`" will be assigned to location entities following $\mathcal{M}(\texttt{LOC}) = \texttt{location}$. These CWs inherently contain the general semantic information of related entity classes and can avoid biases from limited labeled data.

In a specific few-shot task, COPNER maps each entity class $c_i \in C$ into a CWs v_i by $\mathcal{M}$ and concatenates these CWs to form a prompt. There are three ways of concatenating CWs in COPNER, which are described in Section 3.5.

Next, the generated prompt is appended to each input sentence X to form the extended input sequence $X' = \{x_1, x_2, \ldots, x_t, v_1, \ldots, v_n, v_{n+1}\}$[1], where t is the length of original input sentence and n is the number of entity class. Additionally, an extra class-specific word is added to denote non-entity class. Following, X' is fed into a PLM, e.g., BERT (Devlin et al., 2019), for generating contextualized representations. COPNER takes final hidden layer output as the representations of each token following:

$$\begin{aligned}\mathbf{H} &= [\mathbf{h}_1, ..., \mathbf{h}_t, \mathbf{h}'_1, ..., \mathbf{h}'_n] \\ &= \texttt{PLM}([x_1, .., x_t, v_1, ..., v_n])\end{aligned} \tag{1}$$

The representations of CWs in the prompt can be treated as guides for training the representations of tokens in the original sentence, which will detailed in the next section.

3.2 Training in Source Label Space

We employ an episode training strategy (Ding et al., 2021b) in COPNER, where a Greedy Sampling is adopted to randomly select an episode set $\mathcal{S}$ at each step. Specifically, the samples in an episode set $\mathcal{S}$ contain N entity classes (N way) with 1$\sim$2 examples per class.

Centered with CWs in the prompt as class anchors, COPNER trains the used PLM to reduce representation distances between each token with its related CWs while pulling way with unrelated CWs. Specifically, for each extended input sequence $X' \in \mathcal{S}$, COPNER obtains the representation sequence $[\mathbf{h}_1, ..., \mathbf{h}_t, \mathbf{h}'_1, ..., \mathbf{h}'_{n+1}]$ by Eq. (1).

[1]For clarity, our formulates exclude the special marker "[BOS]" and "[EOS]". The detail utilization of these markers is shown in Fig 1.

Algorithm 1 Adapting process of COPNER

```
Input: $\mathcal{X}_{sup}$: support data; PLM: word encoder; $\gamma$: loss threshold;
Output: PLM
 1: $\mathcal{L}_{prev} \in \mathbb{R}_+$ (arbitrary large);
 2: $\mathcal{L}_{ft} = \mathcal{L}_{prev} - 1$;
 3: repeat
 4:     $\mathcal{L}_{prev} = \mathcal{L}_{ft}$;
 5:     for all $(x_i, y_i) \in \mathcal{X}_{sup}$ do
 6:         Calculate $l(x_i)$ as in Eq. (1) and Eq. (2);
 7:     end for
 8:     Calculate $\mathcal{L}_{ft}$ as in Eq. (3);
 9:     Update PLM by back-propagation to reduce $\mathcal{L}_{ft}$;
10: until ($\mathcal{L}_{ft} > \mathcal{L}_{prev} \cap \mathcal{L}_{ft} < \gamma$)
```

Next, we construct positive and negative pairs of each X' for contrastive learning. Positive pairs are defined as (x_p, v_p), where x_p is the p_{th} token in X and v_p is the corresponding gold CW. Negative pairs are obtained by combining x_p and other unrelated CWs in the prompt.

Then, we can calculate the contrastive loss (Lin et al., 2021) with respect to x_p by:

$$\ell(x_p) = -\log \frac{\exp(-d(\mathbf{h}_p, \mathbf{h}'_p)/\tau)}{\sum_{q=1}^{n+1} \exp(-d(\mathbf{h}_p, \mathbf{h}'_q)/\tau)} \tag{2}$$

where τ denotes a temperature hyper-parameter proposed by Chen et al. (2020). As in previous metric-based works, we adopt the Euclidean distance as the similarity measure, which is calculated by: $d(\mathbf{h}_p, \mathbf{h}'_q) = ||\mathbf{h}_p - \mathbf{h}'_q||_2^2$. The total contrastive loss $\mathcal{L}$ of the episode $\mathcal{S}$ is calculate by:

$$\mathcal{L} = \frac{1}{|\mathcal{X}|} \sum_{x_i \in \mathcal{X}} \ell(x_i) \tag{3}$$

where $\mathcal{X}$ denotes the text token set in a sampled $\mathcal{S}$.

3.3 Adapting to Target Label Space

As mentioned in Section 2.1 and 2.2, when handling the settings of Cross-Label-Space and Domain Transfer, COPNER needs a certain extent of transferring ability. For such a reason, COPNER is fine-tuned with related support sets to adapt new label spaces after the training phase. This procedure is similar to the training stage, and the only difference is that the used data come from different label spaces or domains. Noticeably, such adapting process may make COPNER over-fit with the used adapting data, because these adapting data are usually with small numbers. Inspired by Das et al. (2021), we develop an early stopping criterion based on contrastive losses to alleviate the above problem. In particular, we add a hyper-parameter

γ as the loss threshold to prevent the model from not adapting enough or over-fitting. The complete adapting process with the early stopping criterion is illustrated in Algorithm 1.

3.4 Inferring from Metric Referents

In the inference phase, CWs in the prompt are regarded as metric referents to calculate distance with each token. We first obtain the representation of a extended test instance H^{test} following the rewritten Eq. (1) as

$$H^{test} = [\mathbf{h}_1, ..., \mathbf{h}_t, \mathbf{h}'_1, ..., \mathbf{h}'_n, \mathbf{h}'_{n+1}]. \quad (4)$$

For each token x_i, COPNER can find the nearest Metric Referent v_j in the PLM representation space, and the corresponding label c_j will be assigned with this token.

$$y_i^{test} = \underset{c_j}{\text{argmin}} ||\mathbf{h}_i - \mathbf{h}'_j||_2^2 \quad (5)$$

where $\mathbf{h}_i$ denotes each token representation and $\mathbf{h}'_j$ denotes each CW representation.

Alternatively, COPNER employs the Viterbi decoding algorithm. The used transition probabilities are calculated between three abstract NER tags (`O`, `I`, `I-Other`) on the training data. The emission probabilities are calculated by a `SoftMax` operation on the distance distributions between each test token and CWs during inference. These two probabilities are fed to a Viterbi decoder to obtain the final prediction. For more details, please refer to Structshot (Yang and Katiyar, 2020).

3.5 Prompt Construction

Liu et al. (2021b) shows that different forms of prompts have different effects on prompt-based approaches. For further explore these effects on COPNER, we propose three prompt construction methods. Figure 2 shows the examples for these methods and the details are introduced as:

- **Queue Prompt**: Directly combining the CWs in random order (the most intuitive way).
- **Partition Prompt**: Based Queue Prompt, extra special tokens "`[S]`" are used to separate each CWs. "`[S]`" only serves as a partition and does not have a specific meaning.
- **Continual Prompt**: This method employs continuous representations as special tokens to separate CWs. Similar with P-tuning (Liu et al., 2021b), the employed continuous representations are generated from an independent prompt encoder (two Bi-LSTM layers). By this way, we try to encode hidden associations between CWs into these separated markers.

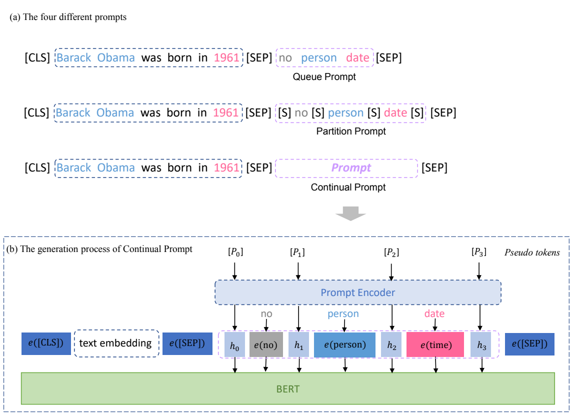

(a) The four different prompts
[CLS] Barack Obama was born in 1961 [SEP] no person date [SEP]
Queue Prompt
[CLS] Barack Obama was born in 1961 [SEP] [S] no [S] person [S] date [S] [SEP]
Partition Prompt
[CLS] Barack Obama was born in 1961 [SEP] Prompt [SEP]
Continual Prompt
(b) The generation process of Continual Prompt
[P0] [P1] [P2] [P3]
Pseudo tokens
Prompt Encoder
no
person
date
e([CLS])
text embedding
e([SEP])
h0 e(no) h1 e(person) h2 e(time) h3
e([SEP])
BERT

Figure 2: The illustration of prompt construction: (a) Three different forms of the prompt. (b) The process of generating continual prompts. A Prompt Encoder is employed to generate continual prompts, which is optimized with COPNER during training. $e(x)$ denotes the embedding of token x.

4 Experiment Setups

4.1 Datasets

COPNER is evaluated with six datasets, including OntoNotes 5.0 (Weischedel et al., 2012), WNUT'17 (Derczynski et al., 2017), I2B2'14 (Stubbs and Uzuner, 2015), CONLL'03 (Sang and De Meulder, 2003), MIT-Movie (Liu et al., 2013), and Few-NERD (Ding et al., 2021b). Among these datasets, the first five datasets come from the different domains, which correspond to the fields of general, social network, medical, newswire, and review, respectively. The last Few-NERD is the largest few-shot NER dataset, which contains INTRA and INTER two sub-settings, and a total of 66 fine-grained classes across 8 coarse-grained categories. Details of the datasets are shown in Table 6.

4.2 Baselines

(1) ProtoBERT is a popular few-shot method based on the prototypical network (Snell et al., 2017) with BERT (Devlin et al., 2019) as a backbone. **(2) NNShot** (Wiseman and Stratos, 2019) is a simple method based on token-level nearest neighbor classification. **(3) StructShot** (Yang and

Katiyar, 2020) adopts an additional Viterbi decoder based on NNShot. **(4) CONTaiNER** (Das et al., 2021) leverages contrastive learning to infer the distributional distance of Gaussian embeddings of entities. **(5) BERT-tagger** (Devlin et al., 2019) is a traditional BERT-based method which fine-tunes the BERT model with a label classifier. **(6) TemplateNER** (Cui et al., 2021) is a template-based approach, which enumerates all possible n-gram spans and classifies each of them. **(7) EntLM** (Ma et al., 2021) is a few-shot NER method which leverages an entity-oriented LM objective.

4.3 Evaluation on Three Settings

Cross-Label-Space Setting. For this setting, we evaluate COPNER with the Few-NERD dataset, which has two different tasks: Few-NERD (INTER) and Few-NERD (INTRA). For INTER, all the fine-grained entity classes are mutually disjoint in train, development, and test sets, while the coarse-grained categories are shared. For INTRA, the fine-grained entity classes in different sets belong to different coarse-grained categories. Few-NERD (INTRA) is more challenging due to the restrictions of sharing coarse-grained types. We evaluate COPNER 5000 episodes on the test set under each setting. As shown in Table 1 and Table 2, COPNER largely outperforms present related SOTAs in both tasks.

Domain Transfer Setting. This setting focuses on transferring an NER model to a new domain. Specifically, we train COPNER on the OntoNotes 5.0 dataset from the general domain and evaluate it on the test sets of CoNLL'03, WNUT'17, I2B2'14, which are from the newswire, social and medical fields, respectively. The support sets of the target domains are provided by Yang and Katiyar (2020). For each experiment, COPNER is fine-tuned on five support sets and the mean and standard deviation of F1 scores on the test set is reported. As shown in Table 3, COPNER also outperforms SOTAs in most cases, especially in the 1-shot setting.

In-Label-Space Setting. In this setting, we evaluate COPNER with two NER datasets from different domains: CoNLL'03 (Sang and De Meulder, 2003) and MIT-Movie (Liu et al., 2013). As introduced in Section 2.3, only K examples of each class are available for training. To explore the few-shot capability of COPNER with different size of training data, we try different K values from $\{5, 10, 20, 50\}$. For each K-shot values, COPNER

Table 1: F1 Scores (%) in Few-NERD (INTER). +Struct means using Viterbi Decoding. We color code each column as best and second best.

Model	5-way		10-way		Avg.
	1∼2 shot	5∼10 shot	1∼2 shot	5∼10 shot	
ProtoBERT	44.44	58.80	39.09	53.97	49.08
NNShot	54.29	50.56	46.98	50.00	50.46
StructShot	57.33	57.16	49.46	49.39	53.34
CONTaiNER	55.95	61.83	48.35	57.12	55.81
+Struct	56.10	61.90	48.36	57.13	55.87
COPNER	65.39	67.59	59.69	62.32	63.75
+Struct	65.98	67.70	59.56	62.37	63.90

Table 2: F1 Scores (%) in Few-NERD (INTRA). +Struct means using Viterbi Decoding. We color code each column as best and second best.

Model	5-way		10-way		Avg.
	1∼2 shot	5∼10 shot	1∼2 shot	5∼10 shot	
ProtoBERT	23.45	41.93	19.76	34.61	29.94
NNShot	31.01	35.74	21.88	27.67	29.08
StructShot	35.92	38.83	25.38	26.39	31.63
CONTaiNER	40.43	53.70	33.84	47.49	43.87
+Struct	40.40	53.71	33.82	47.51	43.86
COPNER	53.52	58.74	44.13	51.55	51.99
+Struct	54.26	58.84	44.26	51.18	52.14

is trained on three different sampled train sets provided by Ma et al. (2021), the mean and standard deviation of F1 scores on the test set is reported. As shown in Table 4, COPNER also outperforms state-of-the-art methods in most cases.

5 Results and Discussion

In this section, we discuss the results of different few-shot NER settings and conduct experiments under the zero-shot setting. We also explore the effectiveness of different components of COPNER.

5.1 Overall Few-shot Results

The experimental results demonstrate that COPNER achieves a convincing improvement in all mentioned few-shot NER settings, and reaches SOTA performance to our best knowledge.

As shown in Table 1 and 2, COPNER outperforms the baselines by a large margin in the Cross-Label-Space setting. We observe a significant improvement in the 1-shot setting. A 1-shot sample may not give sufficient information about the target class distribution, which limits the performance of previous methods to a large extent. In contrast, CWs in COPNER carry class-related semantic information to ensure excellent performance.

As shown in Table 3, COPNER demonstrates strong domain transfer capability where support data are extremely limited. COPNER performs

Table 3: F1 scores (%) in Domain Transfer task. We report standard deviations from runs with five different support sets sampled by Yang and Katiyar (2020). +Struct means using Viterbi Decoding. We color code each column as best and second best .

Model	1 shot				5 shot			
	CoNLL	WNUT	I2B2	Avg.	CoNLL	WNUT	I2B2	Avg.
ProtoBERT	49.9±8.6	17.4±4.9	13.4±3.0	26.9	61.3±9.1	22.8±4.5	17.9±1.8	34.0
NNShot	61.2±10.4	22.7±7.4	15.3±1.6	33.1	74.1±2.3	27.3±5.4	22.0±1.5	41.1
StructShot	62.4±10.5	24.2±8.0	21.4±3.8	36.0	74.8±2.4	30.4±6.5	30.3±2.1	45.2
CONTaiNER	57.8±10.7	24.2±2.9	16.4±1.7	32.8	72.8±2.0	27.7±2.2	24.1±1.9	41.5
+Struct	61.2±10.7	27.5±1.9	21.5±1.7	36.7	75.8±2.7	32.5±3.8	36.7±2.1	48.3
COPNER	67.0±3.8	33.8±2.5	34.6±1.8	45.1	74.9±2.9	34.8±3.1	41.1±1.6	50.2
+Struct	66.5±2.1	34.9±1.8	35.8±1.3	45.7	74.6±3.1	34.2±2.6	43.7±1.5	50.8

Table 4: F1 scores (%) in In-Label-Space NER task. We report standard deviations from runs with three different support sets sampled by Ma et al. (2021). +Struct means using Viterbi Decoding. We color code each column as best and second best .

Model	CONLL					MIT-Movie				
	5 shot	10 shot	20 shot	50 shot	Avg.	5 shot	10 shot	20 shot	50 shot	Avg.
BERT-tagger	41.9±12.1	59.9±10.7	68.7±5.1	73.2±3.1	60.9	39.6±6.4	50.6±7.3	59.3±3.7	71.3±3.0	55.2
NNShot	42.3±8.9	59.2±11.7	66.9±6.1	72.6±3.4	60.3	39.0±5.5	50.5±6.1	58.9±3.5	71.2±2.9	54.9
StructShot	45.8±10.3	62.4±11.0	69.5±6.5	74.7±3.1	63.1	41.6±9.0	53.2±5.5	61.4±3.0	72.0±6.4	57.1
TemplateNER	43.0±6.2	57.9±5.7	66.4±6.1	72.7±2.1	60.0	46.0±3.9	49.3±3.4	59.1±0.4	65.1±0.2	54.9
EntLM	49.5±8.3	64.8±3.9	69.5±4.5	73.7±2.1	64.4	46.6±9.5	57.3±3.7	62.4±4.1	71.9±1.7	59.6
+Struct	51.3±7.7	66.9±3.0	71.2±3.9	74.8±1.9	66.1	49.2±8.9	59.2±4.0	63.9±3.7	73.0±1.8	61.3
COPNER	54.9±4.1	65.3±2.4	70.7±1.8	75.0±1.5	66.5	50.9±4.4	59.7±0.4	66.7±1.8	73.8±0.6	62.8
+Struct	54.2±7.9	66.2±2.9	71.8±1.8	77.0±1.4	67.3	50.1±3.6	61.9±1.4	68.9±2.4	74.6±0.3	63.9

significantly better than all previous methods, especially in the 1-shot setting. Specifically, COPNER raises of 5.8%, 7.4% and 14.3% F1 scores on the CoNLL, WNUT and I2B2 datasets, respectively.

As shown in Table 4, COPNER still achieves SOTA performance in most cases. In the In-Label-Space setting, the generalization ability of COPNER is examined by limiting the available training samples. Additionally, the standard deviations of F1 scores reported by COPNER are lower than those of other baselines, which indicates that our method is more stable than these baselines.

5.2 Zero-shot Learning

After trained on a rich-resource dataset, COPNER has learnt hidden contextual associations between CWs and tokens of input sentences. We aruge that these learnt contextual associations help COPNER to classify unseen entity categories even without any support data. Several experiments are conducted to demonstrate this idea. Specifically, for the Cross-Label-Space task and the Domain Transfer task, COPNER needs to make predictions on the target label spaces without any support set after training on the source label spaces. As shown in Table 5, COPNER can handle zero-shot NER tasks

Table 5: F1 Scores (%) in Zero-shot setting. +Struct means using Viterbi Decoding. We color code each column as best .

Model	Few-NERD				Domain Transfer		
	INTER		INTRA		CONLL	WNUT	I2B2
	5 way	10 way	5 way	10 way	-	-	-
COPNER	31.95	19.52	14.72	8.73	46.26	17.58	17.29
+Struct	33.97	20.92	16.06	9.64	49.39	17.41	17.47

with satisfied performance and the Viterbi decoding can further boost the performance. In the domain transfer tasks, COPNER under the zero-shot setting is even comparable to the prototypical network under the 1-shot setting.

5.3 Effect of Class-specific Words

Considering that the semantic information contained in class names can benefit entity encoding, the class names are selected as CWs introduced in Section 3.1. To demonstrate the effect of semantics of class names, we conduct experiments with the following variants of the CWs:

- Misleading words: We randomly swap the CWs between labels. For example, we assign "`location`" for "`PER`", "`person`" for "`ORG`", "`organization`" for "`LOC`".

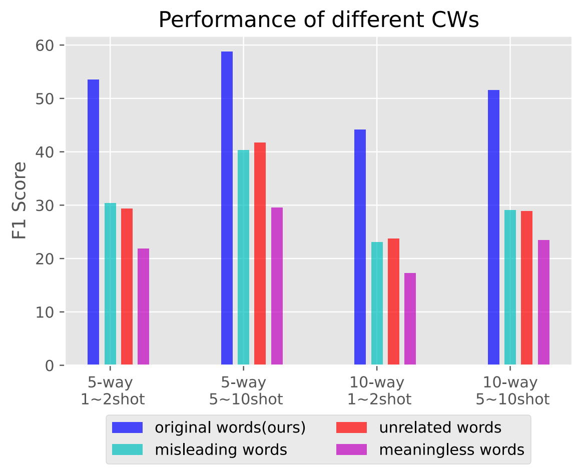


Figure 3: F1 Scores (%) in Few-NERD (INTRA) with the different variations of the CWs.

- Unrelated words: We randomly select some tokens from BERT's vocab as CWs, e.g., "`fully`", "`acoustic`" or "`new`", which are semantically unrelated to labels.

- Meaningless words: We use the tokens that are not used in BERT's vocab for CWs, such as "`[unused0]`", "`[unused1]`", etc. These tokens are semantically meaningless.

As shown in Figure 3, the performance of the three variants shows a significant decrease compared to the original words. The wrong semantic information in misleading words and unrelated words may mislead the entity representation learning leading to huge performance loss. It demonstrates that semantics matching the entity class is more effective as a class anchor. The further decrease in the performance of meaningless words further shows that semantic information is crucial in few-shot metric learning.

5.4 Influence of Prompts

In this subsection, we explore the influences of introducing prompts. In COPNER, prompts provide category-specific information during the representation calculation of tokens. To explore its impact, we conduct analytical experiments and the results are shown in Table 7 in Appendix. In addition to comparing the impact of three different prompt forms introduced in Section 3.5, we construct a baseline without adding any prompts: Fixed embedding Guiding (FG). More details are described in Appendix C. All three prompt-based methods are much better than FG, which indicates that the introduction of prompts can effectively improve the model capability. Both Partition Prompt and Continue Prompt achieve excellent performance, but the latter introduces additional parameters, so we use Partition Prompt in our main experiments.

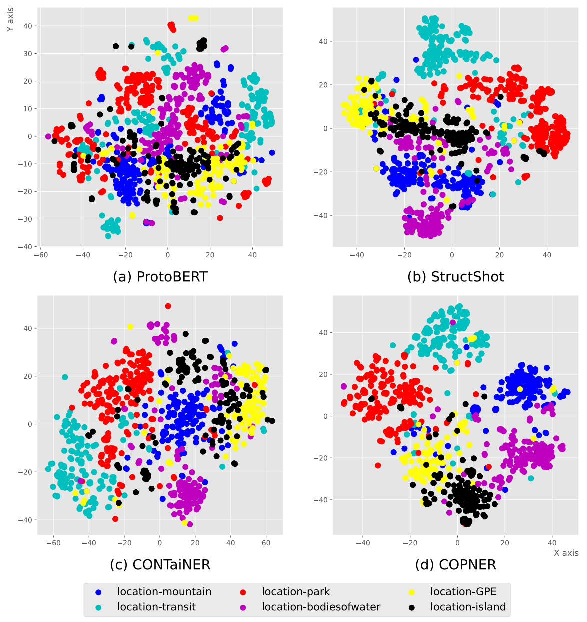


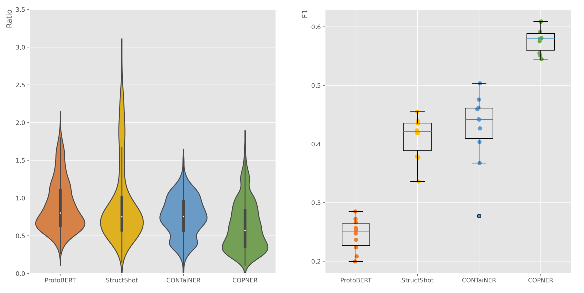
Figure 4: Two-dimensional t-SNE visualizations of the sampled 6 fine-grained classes from the `location` category. The embeddings are obtained from ProtoBERT, StructShot, CONTaiNER and COPNER, respectively.

5.5 Effectiveness Analysis

In this subsection, we summarize two main aspects to show the effectiveness of COPNER.

Enhanced Entity Representations. Figure 4 shows the two-dimension t-SNE visualization for the embeddings obtained from four different metric-based methods. Another one-dimension t-SNE visualization is shown as Figure 6 in Appendix D. As shown as these two Figures, COPNER results in better entity representations with greater differentiation of entity distributions across categories and more aggregation of similar entity distributions. More details are described in Appendix D.

Stable Metric Referents. We further investigate the metric results during inference. As shown in Figure 5(a), COPNER is more capable of distinguishing the positive pairs from the negative pairs with lower positive-negative distance ratios, which indicates that the nearest CW inference in COPNER has stronger category discrimination ability. COPNER is also the most stable and least influenced by support set differences, as shown in Figure 5(b). More details are described in Appendix E.



(a) positive-negative distance ratio (b) F1 scores with 10 different support sets

Figure 5: Effect of Metric Inference. (a) The ratio of the positive-pair distance to the mean negative-pair distance in inference of different models, (b) F1 scores of different models on 10 different support sets.

6 Related Work

Prompt Learning. GPT-3 (Brown et al., 2020) is the first try that used manual prompts in task prediction to improve performance. Then some ways to build templates manually were proposed (Petroni et al., 2019; Sanh et al., 2021). Automatic construction with different forms of templates was explored, including discrete templates (Shin et al., 2020; Jiang et al., 2020; Gao et al., 2020) and continuous prompts (Li and Liang, 2021; Liu et al., 2021b,a; Qin and Eisner, 2021; Han et al., 2021). A proper prompt can provide guidance information for downstream tasks. Therefore, the prompt is well suited for few-shot tasks where training data are scarce. A number of works introducing prompt learning to few-shot classification tasks have been proposed (Mao et al., 2018; Ding et al., 2021a; Chen et al., 2021; Madotto et al., 2021) in succession.

Few-shot NER. Generally, few-shot NER approaches can be categorized as metric-based and prompt-based. The former aims to calculate the similarity between test data and referents. Fritzler et al. (2019) applied the prototype network (Snell et al., 2017) to few-shot NER tasks. Inspired by the nearest neighbor inference (Wiseman and Stratos, 2019), Yang and Katiyar (2020) proposed Structshot, which used the Viterbi Decoder to capture label dependencies. Das et al. (2021) proposed CONTaiNER, which adopts Gaussian embeddings of tokens for the metric. Prompt-based methods leverage prompt learning to exploit the prior knowledge of pre-trained language models. Cui et al. (2021) proposed a time-consuming template-based BART for few-shot NER. Inspired by prompt tuning, Ma et al. (2021) proposed an entity-oriented method that fine-tuned the language model to predict class-related label words rather than the original words.

7 Conclusion

In this paper, we propose COPNER, a novel few-shot NER approach taking the advantage of contrastive learning and prompt guiding. COPNER achieves SOTA performance in few-shot NER settings by constructing prompts with CWs and exploiting the ability of contrastive learning to obtain enhanced representations and stable metric referents. COPNER can also handle zero-shot NER tasks. In the future, we will extend COPNER to more token-level few-shot classification tasks and further exploit its ability to handle zero-shot tasks.

Acknowledgements

This work has been supported by the Innovative Research Group of the National Natural Science Foundation of China (61721002); The Key Research and Development Program of Ningxia Hui Nationality Autonomous Region (2022BEG02025); The Key Research and Development Program of Shaanxi Province (2021GXLH-Z-095); The consulting research project of the Chinese Academy of Engineering (The Online and Offline Mixed Educational Service System for The Belt and Road Training in MOOC China); Project of China Knowledge Centre for Engineering Science and Technology; The innovation team from the Ministry of Education (IRT_17R86).

References

Tom B Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, et al. 2020. Language models are few-shot learners. *arXiv preprint arXiv:2005.14165*.

Ting Chen, Simon Kornblith, Mohammad Norouzi, and Geoffrey Hinton. 2020. A simple framework for contrastive learning of visual representations. In *International conference on machine learning*, pages 1597–1607. PMLR.

Xiang Chen, Ningyu Zhang, Xin Xie, Shumin Deng, Yunzhi Yao, Chuanqi Tan, Fei Huang, Luo Si, and Huajun Chen. 2021. Knowprompt: Knowledge-aware prompt-tuning with synergistic optimization for relation extraction.

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

Leyang Cui, Yu Wu, Jian Liu, Sen Yang, and Yue Zhang. 2021. Template-based named entity recognition using bart. *arXiv e-prints*, pages arXiv–2106.

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*.

Leon Derczynski, Eric Nichols, Marieke van Erp, and Nut Limsopatham. 2017. Results of the wnut2017 shared task on novel and emerging entity recognition. In *Proceedings of the 3rd Workshop on Noisy User-generated Text*, pages 140–147.

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.

Ning Ding, Yulin Chen, Xu Han, Guangwei Xu, Pengjun Xie, Hai-Tao Zheng, Zhiyuan Liu, Juanzi Li, and Hong-Gee Kim. 2021a. Prompt-learning for fine-grained entity typing. *arXiv preprint arXiv:2108.10604*.

Ning Ding, Guangwei Xu, Yulin Chen, Xiaobin Wang, Xu Han, Pengjun Xie, Hai-Tao Zheng, and Zhiyuan Liu. 2021b. Few-nerd: A few-shot named entity recognition dataset. *arXiv preprint arXiv:2105.07464*.

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.

Tianyu Gao, Adam Fisch, and Danqi Chen. 2020. Making pre-trained language models better few-shot learners. *arXiv preprint arXiv:2012.15723*.

Xu Han, Weilin Zhao, Ning Ding, Zhiyuan Liu, and Maosong Sun. 2021. Ptr: Prompt tuning with rules for text classification. *arXiv preprint arXiv:2105.11259*.

Kai He, Lixia Yao, JiaWei Zhang, Yufei Li, Chen Li, et al. 2021. Construction of genealogical knowledge graphs from obituaries: Multitask neural network extraction system. *Journal of Medical Internet Research*, 23(8):e25670.

Zhengbao Jiang, Frank F Xu, Jun Araki, and Graham Neubig. 2020. How can we know what language models know? *Transactions of the Association for Computational Linguistics*, 8:423–438.

Jing Li, Aixin Sun, Jianglei Han, and Chenliang Li. 2020. A survey on deep learning for named entity recognition. *IEEE Transactions on Knowledge and Data Engineering*, 34(1):50–70.

Xiang Lisa Li and Percy Liang. 2021. Prefix-tuning: Optimizing continuous prompts for generation. *arXiv preprint arXiv:2101.00190*.

Qika Lin, Jun Liu, Lingling Zhang, Yudai Pan, Xin Hu, Fangzhi Xu, and Hongwei Zeng. 2021. Contrastive graph representations for logical formulas embedding. *IEEE Transactions on Knowledge and Data Engineering*.

Jingjing Liu, Panupong Pasupat, Yining Wang, Scott Cyphers, and Jim Glass. 2013. Query understanding enhanced by hierarchical parsing structures. In *2013 IEEE Workshop on Automatic Speech Recognition and Understanding*, pages 72–77. IEEE.

Xiao Liu, Kaixuan Ji, Yicheng Fu, Zhengxiao Du, Zhilin Yang, and Jie Tang. 2021a. P-tuning v2: Prompt tuning can be comparable to fine-tuning universally across scales and tasks. *arXiv preprint arXiv:2110.07602*.

Xiao Liu, Yanan Zheng, Zhengxiao Du, Ming Ding, Yujie Qian, Zhilin Yang, and Jie Tang. 2021b. Gpt understands, too. *arXiv preprint arXiv:2103.10385*.

Ruotian Ma, Xin Zhou, Tao Gui, Yiding Tan, Qi Zhang, and Xuanjing Huang. 2021. Template-free prompt tuning for few-shot ner. *arXiv preprint arXiv:2109.13532*.

Andrea Madotto, Zhaojiang Lin, Genta Indra Winata, and Pascale Fung. 2021. Few-shot bot: Prompt-based learning for dialogue systems. *arXiv preprint arXiv:2110.08118*.

Rui Mao and Xiao Li. 2021. Bridging towers of multi-task learning with a gating mechanism for aspect-based sentiment analysis and sequential metaphor identification. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 35, pages 13534–13542.

Rui Mao, Xiao Li, Mengshi Ge, and Erik Cambria. 2022. MetaPro: A computational metaphor processing model for text pre-processing. *Information Fusion*, 86-87:30–43.

Rui Mao, Chenghua Lin, and Frank Guerin. 2018. Word embedding and WordNet based metaphor identification and interpretation. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1222–1231.

Fabio Petroni, Tim Rocktäschel, Sebastian Riedel, Patrick Lewis, Anton Bakhtin, Yuxiang Wu, and Alexander Miller. 2019. Language models as knowledge bases? In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 2463–2473.

Guanghui Qin and Jason Eisner. 2021. Learning how to ask: Querying lms with mixtures of soft prompts. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 5203–5212.

Erik Tjong Kim Sang and Fien De Meulder. 2003. Introduction to the conll-2003 shared task: Language-independent named entity recognition. In *Proceedings of the Seventh Conference on Natural Language Learning at HLT-NAACL 2003*, pages 142–147.

Victor Sanh, Albert Webson, Colin Raffel, Stephen H Bach, Lintang Sutawika, Zaid Alyafeai, Antoine Chaffin, Arnaud Stiegler, Teven Le Scao, Arun Raja, et al. 2021. Multitask prompted training enables zero-shot task generalization. *arXiv preprint arXiv:2110.08207*.

Taylor Shin, Yasaman Razeghi, Robert L Logan IV, Eric Wallace, and Sameer Singh. 2020. Eliciting knowledge from language models using automatically generated prompts. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 4222–4235.

Jake Snell, Kevin Swersky, and Richard Zemel. 2017. Prototypical networks for few-shot learning. In *Proceedings of the 31st International Conference on Neural Information Processing Systems*, pages 4080–4090.

Amber Stubbs and Özlem Uzuner. 2015. Annotating longitudinal clinical narratives for de-identification: The 2014 i2b2/uthealth corpus. *Journal of biomedical informatics*, 58:S20–S29.

Nikhita Vedula, Nedim Lipka, Pranav Maneriker, and Srinivasan Parthasarathy. 2020. Open intent extraction from natural language interactions. In *Proceedings of The Web Conference 2020*, pages 2009–2020.

Ralph Weischedel, Sameer Pradhan, Lance Ramshaw, Jeff Kaufman, Michelle Franchini, Mohammed El-Bachouti, Nianwen Xue, Martha Palmer, Jena D Hwang, Claire Bonial, et al. 2012. Ontonotes release 5.0.

Sam Wiseman and Karl Stratos. 2019. Label-agnostic sequence labeling by copying nearest neighbors. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 5363–5369.

Vikas Yadav and Steven Bethard. 2018. A survey on recent advances in named entity recognition from deep learning models. In *Proceedings of the 27th International Conference on Computational Linguistics*, pages 2145–2158.

Yi Yang and Arzoo Katiyar. 2020. Frustratingly simple few-shot named entity recognition with structured nearest neighbor learning. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 6365–6375.

Morteza Ziyadi, Yuting Sun, Abhishek Goswami, Jade Huang, and Weizhu Chen. 2020. Example-based named entity recognition. *arXiv e-prints*, pages arXiv–2008.

A Data statistics

In our experiments, we utilize a variety of different NER datasets to fully validate the capability of our proposed approach. A summary of these datasets is given in Table 6.

Table 6: Data statistics

Datasets	Domain	#Class	#Sent	#Entity
OntoNotes	General	18	76.7k	104.2k
WNUT'17	Social	6	5.7k	3.9k
I2B2'14	Medical	23	140.8k	29.2k
CoNLL'03	News	4	20.7k	35.1k
MIT-Movie	Review	12	12.2k	26.6k
Few-NERD	General	66	188.2k	491.7k

B Implementation Details

We use the "bert-base-uncased" pre-trained model as the word encoder in all of our experiments. For model training, the AdamW optimizer is employed with learning rate of 1e-4. We set batch size=16 and the loss temperature τ=0.05. As for prompt construction, we adopt Partition Prompt in our main experiments. The micro-F1 score is selected as the standard evaluation metric in all experiments.

Tagging Scheme For fair comparison, we adopt the IO tagging scheme following previous works, where I-type represents that all of the tokens are inside an entity, and O-type denotes all the other tokens.

C Effect of Prompts

In order to explore the effect of prompts, we add a baseline: Fixed embedding Guiding (FG). Specifically, we first obtain the embedding of each class-specific word from the last layer output of the "bert-base-uncased" pre-trained model. We then let these embeddings guide entity representations by contrastive learning. No prompt is expanded after the input text in FG and the embeddings of CWs are fixed during training. We conduct experiments on FG and the other three different prompts introduced in Section 3.5. The experimental results are shown in Table 7.

All the three prompt-based models outperform FG, indicating that prompts are effective in providing category-related information when models perform entity representation calculations. Despite their excellent results, three prompts have some different effects. The best model is Continual Prompt.

Table 7: F1 Scores(%) in Few-NERD (INTRA) with different prompts: QP, PP and CP mean to use Queue Prompt, Partition Prompt and Continual Prompt, respectively. And FG means to use the fixed embedding guiding. We color code each column as best and second best .

Model	5-way		10-way		Avg.
	1∼2 shot	5∼10 shot	1∼2 shot	5∼10 shot	
FG	43.55	51.85	37.49	48.79	45.42
QP	52.15	57.34	42.79	50.99	50.82
PP	53.52	58.74	44.13	51.55	51.99
CP	53.38	58.81	44.40	51.63	52.06

Whereas, it employs an extra Prompt Encoder to generate semantic linkage representations, which introduces additional training parameters. Partition Prompt achieves comparable performance to Continual Prompt, while introducing no additional parameters. This is the reason that we adopt it in our main experiments. The relatively poor performance of Queue Prompt indicates that putting the CWs together instead of separating them with some tokens has negative impacts on the model performance.

D t-SNE Visualization

From the Few-NERD (INTRA) test set, we randomly sample 100 examples from each of six fine-grained classes in the `location` category. Then, t-SNE is employed to project the entity representations obtained by the word encoder into two-dimensional and one-dimensional spaces. It is clear that the way of guiding entity representations by COPNER is more effective.

Figure 4 shows the two-dimensional visualization results of ProtoBERT, StructShot, CONTaiNER and COPNER, respectively. We can observe that: ProtoBERT has the weakest representation ability and fails to distinguish the entity representations of different classes; StructShot improves but tends to distribute the entity representations of the same class to multiple clusters; CONTaiNER further enhances entity representations, while it is still weak in some classes, such as location-bodiesofwater and location-island. COPNER performs the best and brings similar entities together as much as possible.

Figure 6 shows the one-dimensional visualization results of different entity classes. The higher aggregation of similar entity representations obtained from COPNER further visualizes the superiority of the way CWs guiding entity encoding.

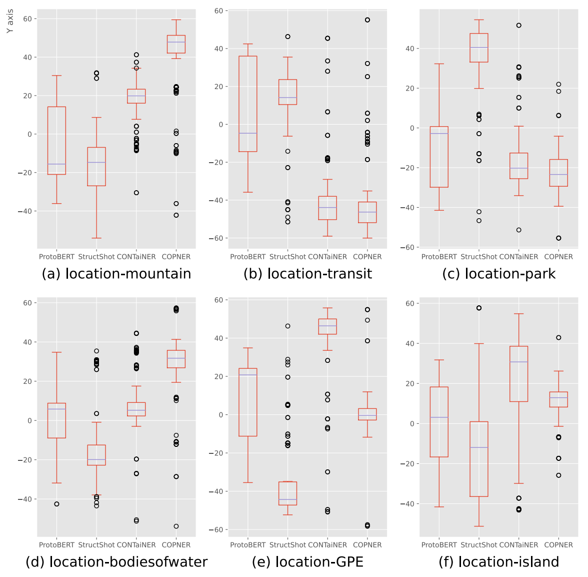


Figure 6: One-dimensional t-SNE visualizations of the sampled 6 fine-grained classes' examples from the `location` category. The 6 different classes are visualized separately.

E Effect of Metric Inference

In this sub-section, we investigate the effect of metric referents for different models during inference. We investigate the following two aspects: category discrimination ability and metric stability.

Category Discrimination Ability: In the distance-based inference, test examples are easily and correctly distinguished when they are closer to the positive metric referent and further away from the negative metric referents. We randomly sample a 1-shot support set containing six fine-grained classes of the location category from the Few-NERD (INTRA) test set, and then make predictions on the test examples sampled in Appendix D. Each test example forms a positive pair with the corresponding gold CW, and negative pairs with other CWs. We calculate the ratio of the positive-pair distance to the mean distance of negative pairs. As shown in Figure 5(a), COPNER is more capable of distinguishing the positive pair from the negative pairs with lower positive-negative distance ratios, which demonstrates that the nearest class-specific word inference in COPNER is more appropriate.

Metric Stability: In the distance-based metric approaches, the inference results of a good metric should be insensitive to different support sets. We randomly sampled 10 different 1-shot support sets from the Few-NERD (INTRA) test set. Each contains 6 fine-grained classes of the location category. Figure 5(b) demonstrates the prediction results of different models. The nearest class-specific word inference in COPNER is the most stable, while the nearest neighbor inference is sensitive to support set differences leading to large differences in prediction results.

F Class-specific Word Examples

Table 8 illustrates several CWs selected by COPNER, which are usually class names with class-specific semantics. For the classes with complex class names, we choose concise words with similar semantics.

Table 8: Several class-specific words selected by COPNER.

Few-NERD		**OntoNotes**	
#Class	#Class-specific word	#Class	#Class-specific word
location-bodiesofwater	water	ORG	organization
location-island	island	NORP	country
person-athlete	athlete	ORDINAL	number
person-director	director	WORK_OF_ART	art
organization-show	show	QUANTITY	quantity
organization-company	company	LAW	law
building-airport	airport	EVENT	event
building-hospital	hospital	CARDINAL	cardinal
art-painting	painting	LOC	location
art-film	film	FAC	facility
...	...	...	...
I2B2		**MIT-Movie**	
#Class	#Class-specific Word	#Class	#Class-specific Word
DATE	date	CHARACTER	character
PATIENT	patient	GENRE	genre
DOCTOR	doctor	TITLE	title
MEDICALRECORD	record	REVIEW	review
HOSPITAL	hospital	RATING	rating
IDNUM	id	YEAR	year
FAX	ip	ACTOR	actor
HEALTHPLAN	plan	DIRECTOR	director
ZIP	zip	SONG	song
STATE	state	RATINGS_AVG	score
...	...	...	...
WNUT		**CoNLL**	
#Class	#Class-specific Word	#Class	#Class-specific Word
location	location	ORG	organization
group	group	PER	person
corporation	company	LOC	location
person	person	MISC	miscellaneous
creative-work	creativity		
product	product		