Coarse-to-fine Few-shot Learning for Named Entity Recognition

Ruotian Ma[1*], **Lin Zhang**[1*], **Xuanting Chen**[1], **Xin Zhou**[1],
Junzhe Wang[1], **Tao Gui**[2†], **Qi Zhang**[1†], **Xiang Gao**[3], **Yunwen Chen**[3]
[1]School of Computer Science, Fudan University, Shanghai, China
[2]Institute of Modern Languages and Linguistics, Fudan University, Shanghai, China
[3]DataGrand Information Technology (Shanghai) Co., Ltd.
{rtma19,zhang_l21,tgui,qz}@fudan.edu.cn

Abstract

Recently, Few-shot Named Entity Recognition has received wide attention with the growing need for NER models to learn new classes with minimized annotation costs. However, one common yet understudied situation is to transfer a model trained with coarse-grained classes to recognize fine-grained classes, such as separating a product category into sub-classes. We find that existing few-shot NER solutions are not suitable for such a situation since they do not consider the sub-class discrimination during coarse training and various granularity of new classes during few-shot learning. In this work, we introduce the Coarse-to-fine Few-shot NER (C2FNER) task and propose an effective solution. Specifically, during coarse training, we propose a cluster-based prototype margin loss to learn group-wise discriminative representations, so as to benefit fine-grained learning. Targeting various granularity of new classes, we separate the coarse classes into extra-fine clusters and propose a novel prototype retrieval and bootstrapping algorithm to retrieve representative clusters for each fine class. We then adopt a mixture prototype loss to efficiently learn the representations of fine classes. We conduct experiments on both in-domain and cross-domain C2FNER settings with various target granularity, and the proposed method shows superior performance over the baseline methods.

1 Introduction

Named Entity Recognition (NER), aiming at recognizing named entities such as person names, locations, and organizations from unstructured text, is a fundamental task for NLP applications (Mintz et al., 2009; Li et al., 2013; Rajpurkar et al., 2016). With the emerging of knowledge from various domains and the emerging of new entity types, it is difficult to manually annotate a large-scale high-quality dataset for each new application scenario (Ding et al., 2021; Yang and Katiyar, 2020). Under this circumstance, Few-shot NER has become a crucial task, which studies NER systems that can fast adapt to new domains or unseen entity types with minimized annotation efforts. Recently, an increasing number of researchers have focused on Few-shot NER and made contributions to this task (Yang and Katiyar, 2020; Huang et al., 2021; Cui et al., 2021; Ma et al., 2022; Tong et al., 2021).

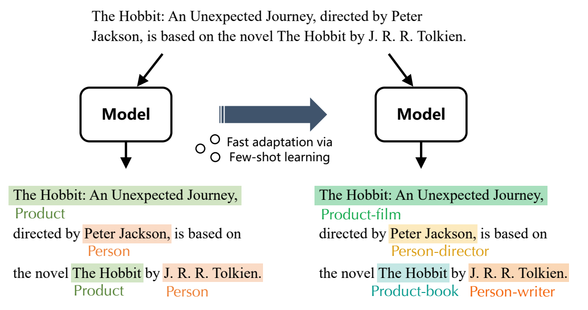
The Hobbit: An Unexpected Journey, directed by Peter Jackson, is based on the novel The Hobbit by J. R. R. Tolkien.
Model
Fast adaptation via Few-shot learning
Model
The Hobbit: An Unexpected Journey, Product directed by Peter Jackson, Person is based on the novel The Hobbit Product by J. R. R. Tolkien. Person
The Hobbit: An Unexpected Journey, Product-film directed by Peter Jackson, Person-director is based on the novel The Hobbit Product-book by J. R. R. Tolkien. Person-writer

Figure 1: A schematic of the Coarse-to-fine Few-shot NER (C2FNER) task. The model trained with coarse annotations is expected to fast adapt to recognizing fine-grained classes with few samples.

Unfortunately, previous works on few-shot NER typically assume the new classes are of the same or similar level of granularity to the known classes (Ding et al., 2021; Yang and Katiyar, 2020). In practice, a common situation is that the model needs to extend to more fine-grained classes from existing coarse-grained classes (Yang et al., 2021; Ni et al., 2021). Such a situation typically occurs during the lifespan of the model when the application requires separating some sub-classes from the current classes and yet when the model is pre-trained without any knowledge of the sub-classes. For example, a product recognizer initially built with a "Computer" class may need to separate into sub-classes such as "Notebook computer" and "Tablet computer" as the requirement updates.

[*]Equal contribution.
[†]Corresponding authors.

Naturally, relabeling a high-quality dataset each time is much too costly to be an option (Bukchin et al., 2021).

In this work, we introduce the Coarse-to-fine Few-shot NER (C2FNER) task. We find that existing few-shot NER solutions are less suitable for C2FNER for two reasons: (i) When training on the coarse-grained classes, most of the existing methods aim at maximizing the inter-class boundaries yet neglect the intra-class variance (Bukchin et al., 2021; Chen et al., 2022b), which might result in reducing separability of sub-classes, thus hindering few-shot fine-grained learning. (ii) As the granularity of new classes is unknown during coarse training, it is challenging to provide a solution that can benefit various granularity of new classes. This requires specific designs for cross-granularity few-shot learning, which is not considered in general few-shot NER methods.

In this work, we propose a novel and effective solution for C2FNER. Firstly, in order to promote the intra-class variance in a way that best benefits cross-granularity learning, we propose a cluster-based prototype margin loss that improves group-wise discrimination in the coarse classes. Specifically, we first perform clustering on the representations of coarse-grained model to obtain entity clusters. Then, we minimize the distance of each entity with its prototype while maximizing its distance with negative prototypes from other coarse classes. Second, in order to benefit various granularity of new classes, we separate each coarse class into extra-fine clusters and propose a novel prototype retrieval algorithm to retrieve representative prototypes (i.e., representative clusters) for each fine-grained classes. By initialize each fine-grained class with multiple prototypes, we then adopt a mixture prototype loss for efficiently learning the representations of fine classes with only few samples. We also bootstrap the prototypes of each fine class with a kNN-based method during few-shot training, so as to make full use of the coarse model.

As the first attempt on C2FNER, we evaluate the proposed method and baselines on various practical C2FNER settings, including (1) the in-domain coarse-to-fine setting where the fine-grained classes are strictly included in and inferior to each coarse-grained class; and (2) the cross-domain coarse-to-fine setting where the fine classes might not strictly included in the coarse classes. Additionally, we evaluate the methods with various granularity of new classes. Experimental results show the superiority of the proposed method over baseline methods on multiple C2FNER settings. [1]

To sum up the contribution of this work:

- In this work, we introduce the Coarse-to-fine Few-shot NER (C2FNER) task, a practical yet understudied task that is hard to solve with existing few-shot solutions.
- We propose a novel and effective method for C2FNER, which not only provide adaptive representations for fine classes during coarse-grained training, but also benefit various granularity of new classes through novel mixture prototypes retrieving and learning.
- We provide comprehensive benchmarks for C2FNER on both in-domain cross-domain settings and for various granularity of new classes.

2 Problem setup

In this work, we introduce the Coarse-to-fine Few-shot NER task (C2FNER). Formally, we denote by $\mathcal{Y}_{coarse} = \{c_1, \dots, c_R\}$ a set of R coarse classes and let $\mathcal{Y}_{fine} = \{c_1, \dots, c_F\}$ be a set of fine classes. In our experiments, we explore both the situations when the fine classes are strictly included or not in the coarse classes. When training on the coarse classes, we assume an adequate number of N training instances $\mathcal{S}^{coarse}_{train} = \{(X_j, Y_j)\}^N_{j=1}$ annotated (only) with $\mathcal{Y}_{coarse}$. When tested on the fine classes, only a K-shot training set $\mathcal{S}^{fine}_{train} = \{(X_j, Y_j)\}^{N_K}_{j=1}$ and the test set $\mathcal{S}^{fine}_{test} = \{(X_j, Y_j)\}^M_{j=1}$ is provided, where K-shot means each fine class is annotated with only K entities. Our goal is to train a model M on the coarse training set $\mathcal{S}^{coarse}_{train}$ and then fine-tune it on the K-shot fine training set $\mathcal{S}^{fine}_{train}$ to obtain maximal performance on the fine test set $\mathcal{S}^{fine}_{test}$.

3 Methodology

As shown in Figure 2, in this work, we propose a novel and effective method for C2FNER, which improves the performance from both the coarse and fine training phases: (1) During coarse training, we propose a cluster-based prototype margin loss to learn group-wise discriminative representations

[1]Our code is publicly available at `https://github.com/rtmaww/C2FNER`.

that benefit fine-grained learning. (2) During fine training, we improve performance for various fine granularity by retrieving representative clusters for each fine class with a prototype retrieval and bootstrapping algorithm and then training with a mixture prototype loss.

3.1 Coarse-grained Representation Learning: Cluster-based Prototype Margin Loss

The goal of coarse training is to provide a fine-friendly representation for future fine-grained adaptation when ensuring the coarse performance. However, general supervised learning or few-shot learning methods aims at maximizing the inter-class margin of coarse classes without considering the sub-classes included in the coarse classes. Such maximization of inter-class variance will harm the intra-class variance, leading to less separable sub-classes and unfriendly representations for coarse-to-fine transfer.

In this work, we propose a cluster-based prototype margin loss for coarse training, which increases the intra-class variance of each coarse class during training as well as learning a good semantic structure of features. Specifically, semantically similar entities are pulled together while being separable from other entities. Moreover, the discrimination of coarse classes should be retained, thus not hindering the performance of the coarse classes. Therefore, we use a prototype for representing a group of semantically similar entities and achieve the above goal with a prototype margin loss. Formally, we first assign the entities in each coarse class c_r to a set of groups $\mathcal{G}_{c_r} = \{g_i\}_{i=1}^{N_G}, \mathcal{G}_{c_r} \subset \mathcal{G}$ by performing clustering on the representations of all entities from c_r. We denote $M(e_i)$ as the assigned group (or cluster) of entity e_i (i.e., $e_i \in M(e_i)$), where $M(\cdot)$ is a mapping function. Then, we obtain the prototype representation $\mathbf{p}_k$ of each group k by:

$$\mathbf{p}_k = \frac{1}{|g_k|} \sum_{e_i \in g_k} \mathbf{h}(e_i) \tag{1}$$

where $\mathbf{h}(e_i)$ is the default representation of e_i from the pre-trained PLM encoder. Then, the prototype margin loss for e_i is calculated by:

$$\begin{aligned} L_{pm,i} = \max(0, m + d(\mathbf{h}(e_i), \mathbf{p}_{M(e_i)}) \\ - \min_{p_n \in \mathcal{G} \setminus \mathcal{G}_{y_i}} d(\mathbf{h}(e_i), \mathbf{p}_n)) \end{aligned} \tag{2}$$

where m is a pre-defined margin. $\mathbf{p}_{\mathcal{M}(e_i)}$ is the assigned prototype of e_i. $\mathbf{p}_n$ is a negative prototype selected from other coarse classes. $d(\cdot)$ denotes a distance function, which we use the Euclidean distance in our implementation. To optimize the prototype margin loss, each entity is forced to get closer to its prototype as well as keep a distance from prototypes from other classes. Consequently, semantically similar entities are pulled together and are discriminative to other groups of entities, thus contributing to friendly representations for coarse-to-fine transfer learning.

As the prototype margin loss is calculated only for entity representations, we also train a classifier with the Cross-Entropy Loss for token classification (including the none-entity class):

$$L_{ce} = -\frac{1}{N} \sum_{i}^{N} \log \frac{\exp(\mathbf{W}_c^\top \mathbf{h}(x_i) + \mathbf{b}_c)}{\sum_{j=1}^{|\mathcal{Y}_{coarse}|} \exp(\mathbf{W}_j^\top \mathbf{h}(x_i) + \mathbf{b}_j)} \tag{3}$$

where $\mathbf{W}_c$ and $\mathbf{b}_c$ is the weight and bias of class c. The final training loss can be written as:

$$L_{coarse} = L_{ce} + \sum_{i}^{N_e} L_{pm,i} \tag{4}$$

where N_e is the entity number in the training set.

3.2 Fast Adaptation for Various Fine-Granularities

One challenge of adapting a coarse model to the fine classes is the unknown granularity of target fine classes. As the model trained on the coarse classes has no prior information about the fine classes, one intuitive solution is to provide adaptive representations that is suitable for various fine granularities. Also, it is essential to make full use of the coarse model and few-shot samples for fast adaptation during the few-shot training phase, which is unexplored neither in previous few-shot methods nor coarse-to-fine few-shot methods.

In this work, we solve the above problem by learning extra-fine prototypes on the coarse classes and performing multi-prototype learning on the fine classes.

3.2.1 Over-clustering for Unknown Fine Granularity

In order to learn adaptive representations for unknown granularity of fine classes, we propose to divide the entities in the coarse classes into extra-fine clusters through over-clustering and represent

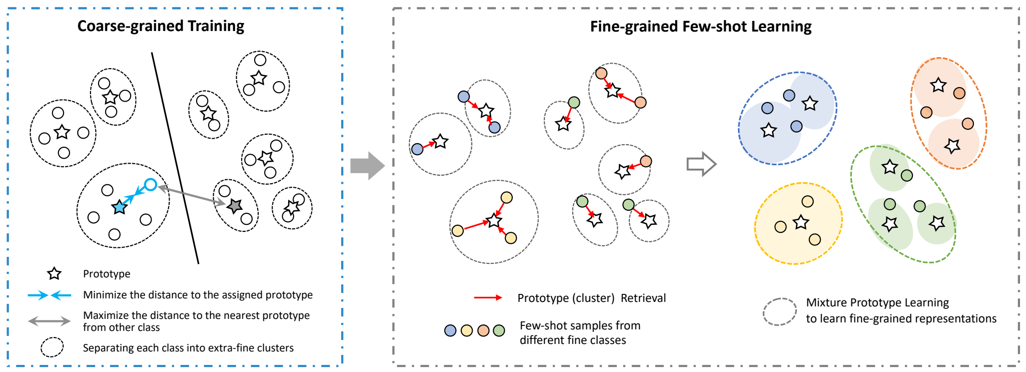
Coarse-grained Training
Prototype
Minimize the distance to the assigned prototype
Maximize the distance to the nearest prototype from other class
Separating each class into extra-fine clusters
Fine-grained Few-shot Learning
Prototype (cluster) Retrieval
Few-shot samples from different fine classes
Mixture Prototype Learning to learn fine-grained representations

Figure 2: An overview of the proposed method. During coarse-grained training, we propose a Cluster-based Prototype Margin Loss to learn group-wise discriminative representations that benefit fine-grained learning. During fine-grained few-shot learning, we propose a Prototype Retrieval algorithm to retrieve representative clusters for each fine class, and then leverage Mixture Prototype Learning to improve fine-grained representations.

each extra-fine cluster with a prototype. As a result, after coarse training, we have:

$$\mathcal{G} = \bigcup_{r=1}^{R} \mathcal{G}_{c_r}, \ |\mathcal{G}| \gg |\mathcal{Y}_{coarse}| \qquad (5)$$
$$\mathcal{P} = \{\mathbf{p}_i\}_{i=1}^{|\mathcal{G}|}$$

By assuming that $|\mathcal{G}| > |\mathcal{Y}_{fine}|$ for most of the fine granularity, the trained representations are adaptive to various fine granularity as long as we perform appropriate training with few-shot samples.

3.2.2 Prototype Retrieval with Few Samples

As the coarse-trained representations are divided into $|\mathcal{G}|$ extra-fine clusters, for arbitrary granularity of fine classes that $|\mathcal{Y}_{fine}| < |\mathcal{G}|$, we propose to retrieve representative clusters for each fine class from the coarse-trained representations. This idea is achieved by a prototype retrieval algorithm. Specifically, for a given fine-grained class $c_f \in |\mathcal{Y}_{fine}|$ with K sampled entities $\{(e_i^f, y_i^f)\}_{i=1}^K$ (where e_i^f denotes the i-th sample of class c_f), we retrieve one prototype for each sample e_i^f by:

$$\hat{P}(e_i^f) = \arg \min_{t \in \{1..|\mathcal{P}|\}} d(\mathbf{h}(e_i), \mathbf{p}_t) \qquad (6)$$

where $\hat{P}(e_i^f)$ is the index of the retrieved prototype, i.e., the retrieved cluster. As the retrieved prototypes of different samples might conflict, it is essential to ensure that each prototype is assigned to only one class. Therefore, for each prototype $\mathbf{p}_j$, we assign it to a possible fine class by:

$$Y(\mathbf{p}_j) = \begin{cases} \text{None}, & \text{if } \forall e_i^f,\ \hat{P}(e_i^f) \neq j \\ \arg\min_{c_f} d(\mathbf{h}(e_i^f), \mathbf{p}_j), & \text{else} \end{cases} \qquad (7)$$

The detailed process of the Prototype Retrieval Algorithm can be found in Alg.1.

Prototype Bootstrapping In order to make the fullest use of the pre-trained representations and discover more representative clusters for each fine class, we also propose to bootstrap the prototypes of each class. To ensure appropriate prototype assignment, we borrow the idea of K-Nearest-Neighbor, i.e., one prototype is most likely to belong to class c_f if most of its nearest neighbors belong to class c_f. Therefore, we bootstrap the prototypes after every M epochs by:

$$\eta_k(c_f \mid \mathbf{p}_i) = \sum_{j=1}^{|\mathcal{P}|} \mathbb{1}[Y(\mathbf{p}_j) = c_f, \mathbf{p}_j \in topK(\mathbf{p}_i)]$$
$$\hat{Y}(\mathbf{p}_i) = \arg\max_{c_f} \eta_k(c_f \mid \mathbf{p}_i) \qquad (8)$$
$$Y(\mathbf{p}_i) = \begin{cases} \hat{Y}(\mathbf{p}_i), & \text{if } \eta_k(\hat{Y}(\mathbf{p}_i) \mid \mathbf{p}_i) > T \\ None, & \text{else} \end{cases}$$

where $\mathbb{1}[\cdot]$ is an indicator function and $topK(\cdot)$ is obtained by selecting the K most nearest neighbor of $\mathbf{p}_i$ by calculating $d(\mathbf{p}_i, \mathbf{p}_l), \mathbf{p}_l \in \mathcal{P}$. Also, as shown in the equation, we assign a prototype to class c_f only when the number of prototypes belonging to c_f is larger than a threshold T, in order to keep the quality of the prototype assignment.

3.2.3 Mixture Prototype Learning

By retrieving prototypes from coarse representations, we initialize the few-shot training of fine classes by regarding each fine class $c_f \in \mathcal{Y}_{\{\}\backslash\rceil}$ as a mixture distribution including multiple sub-clusters represented by multiple prototypes $\{\mathbf{p}_i \mid \mathbf{p}_i \in \mathcal{P}(c_f)\}$. In order for faster adaptation, we make full use of the retrieved prototypes with a Mixture Prototype Loss. Specifically, for each

Algorithm 1 Prototype Retrieval

```
Require: The set of prototypes of the coarse model $\mathcal{P}$, the few-shot sample set of the fine-grained classes $\{\{(e_i^1, y_i^1)\}_{i=1}^K, \ldots, \{(e_i^{|\mathcal{C}_{fine}|}, y_i^{|\mathcal{C}_{fine}|})\}_{i=1}^K\}$.
 1: for each prototype $\mathbf{p}_i \in \mathcal{P}$ do
 2:     // Initialize prototype-class assignments
 3:     $Y(\mathbf{p}_i) \leftarrow$ None
 4:     // Initialize the nearest sample lists
 5:     $NS[i] \leftarrow \varnothing$
 6: end for
 7: // Retrieve the nearest prototype for each sample
 8: for each class $c_f \in \mathcal{Y}_{fine}$ do
 9:     for each entity $e_i^f \in \{e_1^f, \ldots, e_K^f\}$ do
10:         $\hat{P}(e_i^f) \leftarrow$ (Eq.6)
11:         $NS[\hat{P}(e_i^f)] \leftarrow NS[\hat{P}(e_i^f)] \cup \{e_i^f\}$
12:     end for
13: end for
14: // Prototype-class assignment with conflict
15: for each prototype $\mathbf{p}_i \in \mathcal{P}$ do
16:     if $NS[i] \neq \varnothing$ then
17:         $Y(\mathbf{p}_i) \leftarrow \arg\min_{c_f}\{d(\mathbf{h}(e_j^f), \mathbf{p}_i) \mid e_i^f \in NS[i]\}$
18:     end if
19: end for
20: return $\{Y(\mathbf{p}_1), \ldots, Y(\mathbf{p}_{|\mathcal{P}|})\}$
```

entity (e_i, y_i), we estimate its probability to y_i by:

$$p(y_i|e_i) = \frac{\exp(-\sum_{\mathbf{p}_j \in \mathcal{P}(c_i)} d(\mathbf{h}(e_i), \mathbf{p}_j))}{\sum_{k}^{|\mathcal{Y}_{fine}|} \exp(-\sum_{\mathbf{p}_j \in \mathcal{P}(c_k)} d(\mathbf{h}(e_i), \mathbf{p}_j))} \tag{9}$$

Then, the loss is computed as the negative log-likelihood $L_{mixpt} = -\log\ p(y_i|e_i)$ of the true class y_i. In this way, the representations of multiple clusters that belongs to each fine class are forced to gather together and draw closer to the representations of few-shot samples, while become more discriminative to the representations from other clusters. To further adapt the representations, we add an additional prototype distance loss for the prototype learning:

$$L_{pt} = \sum_{k}^{|\mathcal{Y}_{fine}|} \sum_{\substack{\mathbf{p}_i, \mathbf{p}_j \in \mathcal{P}(c_k) \\ i \neq j}} d(\mathbf{p}_i, \mathbf{p}_j) \tag{10}$$

As the mixture prototype loss and prototype distance loss are only calculated for entity representations, we also calculate the Cross-Entropy Loss as in Eq.3. Finally, the training loss can be written as:

$$L_{fine} = L_{ce} + L_{mixpt} + L_{pt} \tag{11}$$

4 Experiments

In C2FNER, the models are first trained on a coarse-grained dataset and then adapted to a fine-grained dataset. To comprehensively evaluate the C2FNER task, we conduct experiments with multiple choice of coarse- and fine-grained settings.

	Datasets	**# Class**	**# Train**	**# Test**
Coarse	*In-domain*			
	Few-NERD (coarse)	8	65.8k	17.9k
	Cross-domain			
	CoNLL 2003	4	14.0k	3.4k
	OntoNotes 5.0	18	60.0k	8.4k
Fine	**Datasets**	**# Class**	**# Train**	**# Test**
	Few-NERD (fine)	66	-	28.2k
	Few-NERD (Medium-fine)	39	-	28.2k

Table 1: Dataset details. **# Class** denotes the number of entity types in each dataset.

Two fine granularity To evaluate the model with unknown fine granularity, we construct datasets of two fine granularity based on the Few-NERD dataset (Ding et al., 2021): (a) **Fine**: containing the original 66 fine-grained classes; (b) **Medium-fine**: we combine similar classes in the 66 classes and construct a new granularity including 38 classes, referred to as the Medium-fine dataset. [2]

Multiple coarse-grained datasets We adopt three widely-used datasets for coarse-grained training, corresponding to two C2FNER settings: (a) **In-domain**: We construct a coarse-grained dataset based on the Few-NERD dataset with the 8 coarse classes hierarchical to the 66 fine-grained classes. (b) **Cross-domain**: We adopt the CoNLL 2003 (Tjong Kim Sang and De Meulder, 2003) and OntoNotes 5.0 (Weischedel et al., 2013) datasets from the newswire domain as the coarse-grained datasets for the cross-domain C2FNER setting.

4.1 Few-shot Experiment Settings

For different coarse- and fine-grained choices, we follow the evaluation settings from (Simple, container) and evaluate the methods with all-way 1-shot and 5-shot support set and the standard test set. For both settings, we sampled 3 different few-shot training sets. For each sampled set, we repeat experiments 3 times. Each reported result is the average result of 3×3 experiments.

Evaluation Metric General NER researches adopt the entity-level micro-f1 score for evaluating the models. However, due to the long-tailed class distribution of the fine-grained test set, evaluating with the micro-f1 score less reflect the performance on the small classes. Therefore, we report both the macro-f1 and micro-f1 scores for a more

[2]The details of the dataset construction are in Appendix A.3.

Methods	Medium-fine (39 entity classes)				Fine (66 entity classes)			
	1-shot		5-shot		1-shot		5-shot	
	(micro-f1)	(macro-f1)	(micro-f1)	(macro-f1)	(micro-f1)	(macro-f1)	(micro-f1)	(macro-f1)
BERT	17.36 ± 3.00	18.09 ± 1.71	37.78 ± 2.86	36.08 ± 2.26	13.22 ± 1.29	14.41 ± 1.66	37.45 ± 2.48	34.38 ± 1.66
ProtoNet	4.98 ± 1.34	5.11 ± 1.77	21.04 ± 4.77	19.28 ± 4.89	5.39 ± 2.25	4.77 ± 1.60	20.72 ± 4.47	19.95 ± 1.61
SCL	18.52 ± 4.48	19.41 ± 3.26	40.87 ± 3.36	38.85 ± 2.42	16.45 ± 3.22	17.43 ± 1.96	39.20 ± 2.26	38.08 ± 1.20
NNShot	13.76 ± 2.28	11.38 ± 1.58	31.72 ± 2.58	30.23 ± 1.23	10.12 ± 2.79	9.08 ± 1.20	28.96 ± 01.79	29.52 ± 0.83
StructShot	15.06 ± 2.40	12.80 ± 1.80	33.53 ± 2.76	31.96 ± 1.33	11.53 ± 2.77	10.33 ± 1.17	30.83 ± 1.82	31.47 ± 0.90
CONTaiNER	12.96 ± 3.67	12.66 ± 2.77	29.45 ± 2.43	25.23 ± 0.83	13.56 ± 2.78	12.21 ± 1.33	24.91 ± 1.46	20.74 ± 1.15
FCDC	17.80 ± 5.36	16.65 ± 3.51	37.74 ± 2.07	36.51 ± 2.18	15.25 ± 3.03	14.12 ± 0.88	38.14 ± 1.25	36.04 ± 1.01
Ours	**28.19 ± 3.89**	**30.80 ± 2.70**	**42.88 ± 3.88**	**42.03 ± 1.66**	**25.89 ± 3.86**	**27.72 ± 1.07**	**40.27 ± 3.03**	**41.62 ± 1.19**

Table 2: In-domain 1-shot and 5-shot C2FNER results of the baseline methods and the proposed method. Each reported result is the average result of 3×3 repeated experiments and the number after ± is the standard deviation. We report both the micro- and macro-f1 results for a more comprehensive evaluation on the long-tailed test set.

Methods	Coarse domain: OntoNotes 5.0				Coarse domain: CoNLL 2003			
	Medium-fine		Fine		Medium-fine		Fine	
	(micro-f1)	(macro-f1)	(micro-f1)	(macro-f1)	(micro-f1)	(macro-f1)	(micro-f1)	(macro-f1)
BERT	34.11 ± 3.15	28.86 ± 2.02	33.73 ± 2.68	31.81 ± 1.31	27.59± 2.65	24.57± 1.45	24.81± 6.03	23.74±5.62
ProtoNet	20.50 ± 4.02	18.82 ± 3.06	19.84 ± 3.81	19.92 ± 3.40	16.13± 5.37	16.69± 2.98	16.09± 2.60	16.27±3.23
SCL	34.28 ± 2.27	29.83 ± 1.77	30.19 ± 1.31	29.45 ± 0.87	26.25± 2.06	23.34± 1.36	25.04± 1.52	24.11±1.31
NNShot	25.11 ± 1.60	21.56 ± 1.00	15.20 ± 2.32	14.83 ± 1.38	18.49± 1.68	16.91± 1.03	18.38± 1.45	18.04±0.82
StructShot	26.48 ± 1.67	23.21 ± 0.97	18.27 ± 2.39	17.53 ± 1.51	19.41± 1.73	17.79± 1.08	19.27± 1.50	19.10±0.83
CONTaiNER	28.74 ± 3.96	24.32 ± 1.75	23.37 ± 1.75	21.25 ± 1.27	15.05± 2.01	15.38± 1.47	14.24± 0.75	15.01±0.62
FCDC	33.06 ± 3.93	28.56 ± 2.64	28.89 ± 2.21	28.44 ± 1.08	29.32± 2.29	26.21± 1.78	27.11± 2.25	26.91±1.17
Ours	**36.50 ± 4.08**	**33.65 ± 2.47**	**35.84 ± 2.73**	**35.42 ± 0.69**	**32.83± 2.11**	**31.87± 1.56**	**32.08± 3.00**	**33.68±0.54**

Table 3: Cross-domain 5-shot C2FNER results of the baseline methods and the proposed method. Each reported result is the average result of 3×3 repeated experiments and the number after ± is the standard deviation.

comprehensive evaluation.

4.2 Baselines

We evaluate the C2FNER task with various baseline methods: [3].

General few-shot methods: We include directly fine-tuning on BERT (Devlin et al., 2019) with a token classifier as a general baseline, referred to as **BERT**. Besides, the Prototypical Network (Snell et al., 2017), denoted as **ProtoNet**, is a typical few-shot learning method. We also include **SCL** (Khosla et al., 2020; Gunel et al., 2020), the supervised contrastive learning method that has shown great advantages in few-shot NLP tasks (Zhang et al., 2021; Li et al., 2021).

Few-shot NER methods: **NNShot** and **StructShot** (Yang and Katiyar, 2020) is two few-shot NER methods based on token-level nearest neighbor classification. **CONTaiNER** (Das et al., 2022) is an up-to-date few-shot NER method that leverages a novel contrastive learning technique.

Coarse-to-fine method: **FCDC** (An et al., 2022) is a recently published research on coarse-to-fine category discovery for text classification. Specifically, it proposes a hierarchical weighted self-contrastive network for learning a fine-friendly representation during the coarse training.

[3]The implementation details are in Appendix A.1

4.3 Main Results

Table 2 and Table 3 show the results of the proposed method and the baseline methods on the C2FNER task on the in-domain and cross-domain settings, respectively. From the results, we can observe that: (1) The few-shot NER methods, although effective on previous few-shot NER settings, are less advantageous on the C2FNER task. This might be because these methods are mainly designed for generalizing to classes of the same or similar level of granularity, thus are less beneficial for coarse-to-fine generalization. (2) Surprisingly, the FCDC method, which is proposed for discovering fine-grained classes with coarse-grained annotations on text classification, is less effective on C2FNER. This might be because the hierarchical self-contrastive method assumes different granularities exist in different layers of BERT, while in token-level tasks like NER, the token-level representations are naturally more fine-grained than the sentence-level representations. (3)

Methods	**Medium-fine (39 entity classes)**				**Fine (66 entity classes)**			
	1-shot		**5-shot**		**1-shot**		**5-shot**	
	(micro-f1)	(macro-f1)	(micro-f1)	(macro-f1)	(micro-f1)	(macro-f1)	(micro-f1)	(macro-f1)
Ours	**28.19** ± **3.89**	**30.80** ± **2.70**	**42.88** ± **3.88**	**42.03** ± **1.66**	**25.89** ± **3.86**	**27.72** ± **1.07**	**40.27** ± **3.03**	**41.62** ± **1.19**
w/o CPML	20.28 ± 1.18	22.15 ± 1.12	40.06 ± 2.62	39.71 ± 1.85	19.11 ± 2.95	19.09 ± 0.81	37.39 ± 2.55	38.96 ± 0.75
w/o PR&MPL	23.57 ± 5.72	24.90 ± 3.30	42.30 ± 2.66	39.74 ± 1.15	20.01 ± 3.20	22.35 ± 1.70	39.20 ± 2.27	38.01 ± 1.33
w/o CPML **w/o** PR&MPL	17.36 ± 3.00	18.09 ± 1.71	37.78 ± 2.86	36.08 ± 2.26	13.22 ± 1.29	14.41 ± 1.66	37.45 ± 2.48	34.38 ± 1.66

Table 4: Ablation study on the proposed method on in-domain C2FNER setting. CPML denotes the Cluster-based Prototype Margin Loss for coarse-grained training. PR&MPL denotes the Prototype Retrieval algorithm and Mixture Prototype Learning for fine-grained few-shot learning.

The proposed method shows significant advantages on both in-domain and cross-domain C2FNER settings. On the in-domain setting, the proposed method achieves up to 12.67% micro-f1 score and 13.31% macro-f1 score improvement on 1-shot learning over the baseline methods. On the cross-domain setting, all the methods show decreased results due to the data distribution shift across domains, among which the proposed method consistently outperforms all baseline methods. (4) It is also worth noting that the proposed method shows more advantages regarding the macro-f1 results, indicating that it might better benefit the long-tailed classes. We further illustrate this phenomenon in Sec. 4.6.

4.4 Ablation Study

We conduct an ablation study on the proposed method, as shown in Tab.4. w/o CPML means we didn't adopt the Cluster-based Prototype Margin Loss during coarse training, and directly performed clustering on BERT and then used Prototype Retrieval and Mixture Prototype Learning. w/o PR&MPL means we didn't adopt the Prototype Retrieval and Mixture Prototype Learning and w/o CPML w/o PR&MPL is the pure BERT model without any of our methods. We can see that both CPML and PR&MPL do contribute to the model improvement. The CPML is able to learn appropriate representations for fine classes. The PR&MPL is more effective regarding the macro-f1 results, and more interesting, it can improve the results limitedly when performing without CPML, since it is hard to retrieve appropriate clusters for fine classes without CPML.

4.5 Effect of the Cluster Number

To train the model with Cluster-based Prototype Margin Loss, we first perform clustering on the coarse classes with K-means clustering. In our implementation, we set the cluster number in each class to 50. Figure 3 shows the effect of different

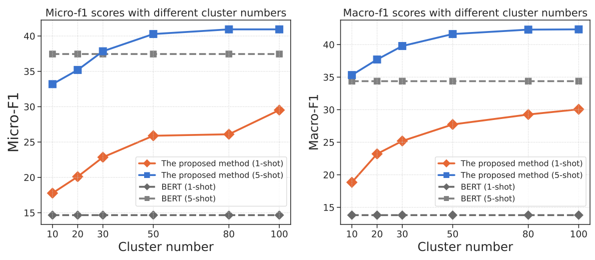

Figure 3: Effect of different cluster numbers.

cluster numbers on the Fine dataset. We can see that the performance grows as the cluster number increases. This might be because the proposed prototype retrieval and mixture prototype learning method are based on regarding each fine class as a mixture distribution, which is ensured by performing over-clustering on the coarse classes. Additionally, when the cluster number is larger than 50, the performance is relatively stable, showing the robustness of the prototype retrieval and mixture prototype learning method towards different over-clustered representations.

Class name	**# Entity**	**Methods**		
		BERT	**SCL**	**Ours**
event-election	131	14.04	16.73	**21.35**
building-restaurant	173	17.34	18.02	**24.88**
product-train	213	27.36	34.63	**37.33**
building-hotel	222	27.23	31.27	**41.50**
building-airport	245	57.12	59.46	**69.90**
other-educationdegree	248	35.68	39.61	**49.10**
product-ship	256	40.50	42.44	**49.96**

Table 5: F1-score on the long-tailed entity classes. **# Entity** denotes the entity number in the test set. Each result is the average result of 3×3 repeated experiments.

4.6 Benefiting Long-tailed Classes

As observed in Table 2 and Table 3, the proposed method is more advantageous regarding the macro-f1 scores, which might be related with the performance of long-tailed classes. In Table 5, we show the results on the smallest classes of different methods. We can see that the proposed method indeed outperforms the baseline methods by a large

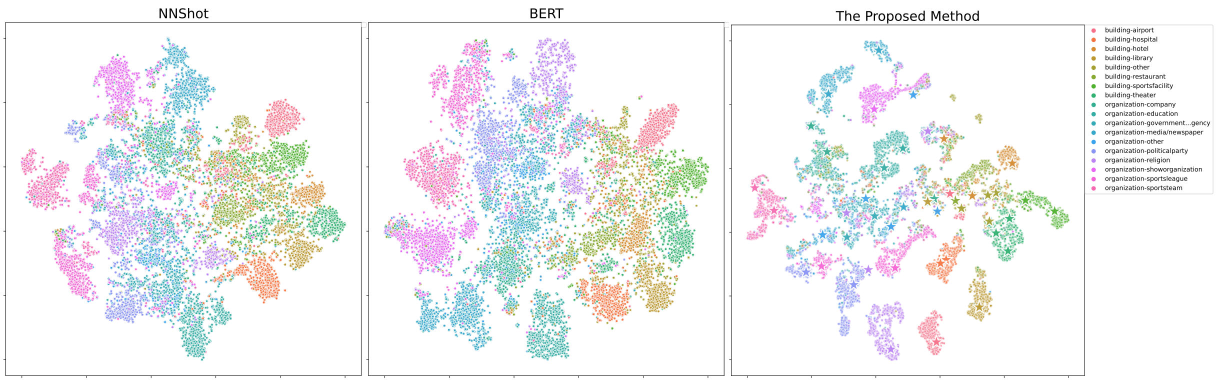

Figure 4: T-SNE visualization of **NNShot** (left), **BERT** (Medium), and **the proposed method** (right). We further annotate the retrieved prototypes ($\star$) of each fine class through Prototype Retrieval. The visualization of more baselines can be seen in Appendix A.2.

margin on these long-tailed classes, which might credit to the over-clustering and Prototype Retrieval method that learn and align small clusters to their corresponding fine classes.

4.7 Visualization of Coarse Representations

In figure 4, we visualize the representations of different models after coarse training. We can observe that both the NNShot and BERT show less separable representations after coarse training due to disregard for the structure of sub-classes. By training with Cluster-based Prototype Margin Loss, we obtain group-discriminative representations that benefit coarse-to-fine learning. Furthermore, we visualize the prototypes ($\star$) retrieved through Prototype Retrieval of each fine class. We can see that the Prototype Retrieval algorithm can retrieve various representative clusters for each fine class with a certain accuracy.

5 Related Works

5.1 Coarse and Fine Learning

The coarse and fine learning problem has been studied in many works. Early works (Ristin et al., 2015; Guo et al., 2018; Taherkhani et al., 2019; Hsieh et al., 2019; Robinson et al., 2020) on coarse-to-fine learning often assume the knowledge of fine classes and even a number of samples of fine classes is available during coarse training, which is less practical to meet the continually-updating user requirements. Recently, several works have explored similar scenarios with C2FNER (Bukchin et al., 2021; An et al., 2022; Yang et al., 2021) and proposed effective solutions. These methods typically focus on learning adaptive representations during coarse training by leveraging contrastive learning (Bukchin et al., 2021; An et al., 2022), meta-learning (Yang et al., 2021) or cluster-based (Ni et al., 2022) methods. In this work, we propose a new solution to learn good representations during coarse training. Moreover, we are the first to consider different target granularity and provide solutions to improve the few-shot learning part.

5.2 Few-shot Learning for NER

Recently, many works have focused on few-shot NER. Yang and Katiyar (2020) improved the prototypical network with a nearest-neighbor-based learning method. Huang et al. (2021) conducted intensive experiments on different methods for few-shot NER. Tong et al. (2021) also leverages a cluster-based method for few-shot NER, yet it only focuses on clustering the none-entity class. More recently, (Das et al., 2022) leverages contrastive learning for few-shot NER. Many works (Cui et al., 2021; Ma et al., 2022; Chen et al., 2022a,c; Huang et al., 2022b) explore better leveraging the pre-trained models with prompt learning or generation-based methods. Generally, these studies only consider few-shot generalization across classes of similar granularity, and are less effective when facing coarse-to-fine situations. To the best of our knowledge, this work is the first to consider and improve coarse-to-fine transfer learning for the NER task.

5.3 Relation with Entity Typing

The proposed C2FNER task is partially related to Entity Typing (Choi et al., 2018; Ma et al., 2016; Del Corro et al., 2015), a task that classifies recognized entities into fine entity types. Recently, several works on few-shot entity typing have achieve great performance on various benchmarks (Huang et al., 2022a; Ding et al., 2022; Eberts et al., 2020). In C2FNER, we aim to adapt a NER model trained on coarse classes to recognize finer classes. When the fine classes are strictly included in the coarse classes, we can simply train a few-shot fine-grained entity typing model to accomplish the coarse-to-fine transfer. However, such practice is restricted to the in-domain setting and cannot be applied to more general situations when the fine classes are not strictly included in the coarse classes. Also, the practice requires the deployment cost of an extra model. Therefore, the proposed method to adapt the NER model is more efficient and more generally applicable.

6 Conclusion

In this work, we introduce the Coarse-to-fine Few-shot NER (C2FNER) task, which considers the practical situation when the models are required to be updated to more fine-grained classes. To solve the C2FNER task, we propose a novel method, which learns fine-friendly representations via a Cluster-based Prototype Margin Loss, and further benefits few-shot learning for various fine granularity via a Prototype Retrieval and Mixture Prototype Learning method. We provide a comprehensive benchmark for C2FNER with different settings, where the proposed method consistently outperforms all baseline methods.

7 Limitations

There are several limitations of this work: (1) In this work, we leverage a cluster-based method for improving the representations during coarse training. Regarding the clustering, we simply perform a K-means process to obtain cluster assignment. The problem is, the cluster assignment with one-time clustering, especially based on the fine-tuned BERT model, might be incorrect. Training with incorrect cluster assignments will pull together the instances from different fine classes, thus hindering the performance. Therefore, the proposed method can be further improved by: (a) leveraging a more robust learning method that can tolerate incorrect cluster assignments; (b) Improving the clustering process to obtain more accurate cluster assignments. (2) In this work, we aim to extend existing few-shot NER to a practical yet understudied setting. However, we limit the setting to the few-shot coarse-to-fine transfer learning setting as in previous works (Bukchin et al., 2021; An et al., 2022; Yang et al., 2021; Ni et al., 2022). In practice, there might be more complex situations that require to be explored in further works.

Acknowledgements

The authors wish to thank the anonymous reviewers for their helpful comments. This work was partially funded by the National Natural Science Foundation of China (No.62076069,62206057,61976056), Shanghai Rising-Star Program (23QA1400200), and Natural Science Foundation of Shanghai (23ZR1403500).

References

Wenbin An, Feng Tian, Ping Chen, Siliang Tang, Qinghua Zheng, and QianYing Wang. 2022. Fine-grained category discovery under coarse-grained supervision with hierarchical weighted self-contrastive learning.

Guy Bukchin, Eli Schwartz, Kate Saenko, Ori Shahar, Rogerio Feris, Raja Giryes, and Leonid Karlinsky. 2021. Fine-grained angular contrastive learning with coarse labels. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 8730–8740.

Jiawei Chen, Qing Liu, Hongyu Lin, Xianpei Han, and Le Sun. 2022a. Few-shot named entity recognition with self-describing networks. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 5711–5722, Dublin, Ireland. Association for Computational Linguistics.

Mayee Chen, Daniel Y Fu, Avanika Narayan, Michael Zhang, Zhao Song, Kayvon Fatahalian, and Christopher Ré. 2022b. Perfectly balanced: Improving transfer and robustness of supervised contrastive learning. In *International Conference on Machine Learning*, pages 3090–3122. PMLR.

Xiang Chen, Lei Li, Shumin Deng, Chuanqi Tan, Changliang Xu, Fei Huang, Luo Si, Huajun Chen, and Ningyu Zhang. 2022c. LightNER: A lightweight tuning paradigm for low-resource NER via pluggable prompting. In *Proceedings of the 29th International Conference on Computational Linguistics*, pages 2374–2387, Gyeongju, Republic of Korea. International Committee on Computational Linguistics.

Eunsol Choi, Omer Levy, Yejin Choi, and Luke Zettlemoyer. 2018. Ultra-fine entity typing. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 87–96, Melbourne, Australia. Association for Computational Linguistics.

Leyang Cui, Yu Wu, Jian Liu, Sen Yang, and Yue Zhang. 2021. Template-based named entity recognition using BART. In *Findings of the Association for Computational Linguistics: ACL-IJCNLP 2021*, pages 1835–1845, Online. Association for Computational Linguistics.

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.

Luciano Del Corro, Abdalghani Abujabal, Rainer Gemulla, and Gerhard Weikum. 2015. FINET: Context-aware fine-grained named entity typing. In *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing*, pages 868–878, Lisbon, Portugal. Association for Computational Linguistics.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. BERT: Pre-training of deep bidirectional transformers for language understanding. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 4171–4186, Minneapolis, Minnesota. Association for Computational Linguistics.

Ning Ding, Yulin Chen, Xu Han, Guangwei Xu, Xiaobin Wang, Pengjun Xie, Haitao Zheng, Zhiyuan Liu, Juanzi Li, and Hong-Gee Kim. 2022. Prompt-learning for fine-grained entity typing. In *Findings of the Association for Computational Linguistics: EMNLP 2022*, pages 6888–6901, Abu Dhabi, United Arab Emirates. Association for Computational Linguistics.

Ning Ding, Guangwei Xu, Yulin Chen, Xiaobin Wang, Xu Han, Pengjun Xie, Haitao Zheng, and Zhiyuan Liu. 2021. Few-NERD: A few-shot named entity recognition dataset. 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 3198–3213, Online. Association for Computational Linguistics.

Markus Eberts, Kevin Pech, and Adrian Ulges. 2020. ManyEnt: A dataset for few-shot entity typing. In *Proceedings of the 28th International Conference on Computational Linguistics*, pages 5553–5557, Barcelona, Spain (Online). International Committee on Computational Linguistics.

Beliz Gunel, Jingfei Du, Alexis Conneau, and Veselin Stoyanov. 2020. Supervised contrastive learning for pre-trained language model fine-tuning. In *International Conference on Learning Representations*.

Yanming Guo, Yu Liu, Erwin M. Bakker, Yuanhao Guo, and Michael S. Lew. 2018. Cnn-rnn: a large-scale hierarchical image classification framework. *Multimedia Tools and Applications*.

John A Hartigan and Manchek A Wong. 1979. Algorithm as 136: A k-means clustering algorithm. *Journal of the royal statistical society. series c (applied statistics)*, 28(1):100–108.

Cheng-Yu Hsieh, Miao Xu, Gang Niu, Hsuan-Tien Lin, and Masashi Sugiyama. 2019. A pseudo-label method for coarse-to-fine multi-label learning with limited supervision.

Jiaxin Huang, Chunyuan Li, Krishan Subudhi, Damien Jose, Shobana Balakrishnan, Weizhu Chen, Baolin Peng, Jianfeng Gao, and Jiawei Han. 2021. Few-shot named entity recognition: An empirical baseline study. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 10408–10423, Online and Punta Cana, Dominican Republic. Association for Computational Linguistics.

Jiaxin Huang, Yu Meng, and Jiawei Han. 2022a. Few-shot fine-grained entity typing with automatic label interpretation and instance generation. In *Proceedings of the 28th ACM SIGKDD Conference on Knowledge Discovery and Data Mining*, pages 605–614.

Yucheng Huang, Kai He, Yige Wang, Xianli Zhang, Tieliang Gong, Rui Mao, and Chen Li. 2022b. COPNER: Contrastive learning with prompt guiding for few-shot named entity recognition. In *Proceedings of the 29th International Conference on Computational Linguistics*, pages 2515–2527, Gyeongju, Republic of Korea. International Committee on Computational Linguistics.

Prannay Khosla, Piotr Teterwak, Chen Wang, Aaron Sarna, Yonglong Tian, Phillip Isola, Aaron Maschinot, Ce Liu, and Dilip Krishnan. 2020. Supervised contrastive learning. *Advances in Neural Information Processing Systems*, 33:18661–18673.

Linyang Li, Demin Song, Ruotian Ma, Xipeng Qiu, and Xuanjing Huang. 2021. Knn-bert: fine-tuning pre-trained models with knn classifier. *arXiv preprint arXiv:2110.02523*.

Qi Li, Heng Ji, and Liang Huang. 2013. Joint event extraction via structured prediction with global features. In *Proceedings of the 51st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 73–82, Sofia, Bulgaria. Association for Computational Linguistics.

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.

Yukun Ma, Erik Cambria, and Sa Gao. 2016. Label embedding for zero-shot fine-grained named entity typing. In *Proceedings of COLING 2016, the 26th International Conference on Computational Linguistics: Technical Papers*, pages 171–180.

Mike Mintz, Steven Bills, Rion Snow, and Daniel Jurafsky. 2009. Distant supervision for relation extraction without labeled data. In *Proceedings of the Joint Conference of the 47th Annual Meeting of the ACL and the 4th International Joint Conference on Natural Language Processing of the AFNLP*, pages 1003–1011, Suntec, Singapore. Association for Computational Linguistics.

Jingchao Ni, Wei Cheng, Zhengzhang Chen, Takayoshi Asakura, Tomoya Soma, Sho Kato, and Haifeng Chen. 2021. Superclass-conditional gaussian mixture model for learning fine-grained embeddings. In *International Conference on Learning Representations*.

Jingchao Ni, Wei Cheng, Zhengzhang Chen, Takayoshi Asakura, Tomoya Soma, Sho Kato, and Haifeng Chen. 2022. Superclass-conditional gaussian mixture model for learning fine-grained embeddings. In *International Conference on Learning Representations*.

Pranav Rajpurkar, Jian Zhang, Konstantin Lopyrev, and Percy Liang. 2016. SQuAD: 100,000+ questions for machine comprehension of text. In *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing*, pages 2383–2392, Austin, Texas. Association for Computational Linguistics.

Marko Ristin, Juergen Gall, Matthieu Guillaumin, and Luc Van Gool. 2015. From categories to subcategories: large-scale image classification with partial class label refinement. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 231–239.

Joshua Robinson, Stefanie Jegelka, and Suvrit Sra. 2020. Strength from weakness: Fast learning using weak supervision. In *Proceedings of the 37th International Conference on Machine Learning*, volume 119 of *Proceedings of Machine Learning Research*, pages 8127–8136. PMLR.

Jake Snell, Kevin Swersky, and Richard Zemel. 2017. Prototypical networks for few-shot learning. *Advances in neural information processing systems*, 30.

Fariborz Taherkhani, Hadi Kazemi, Ali Dabouei, Jeremy Dawson, and Nasser M. Nasrabadi. 2019. A weakly supervised fine label classifier enhanced by coarse supervision. *International Conference on Computer Vision*.

Erik F. 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.

Meihan Tong, Shuai Wang, Bin Xu, Yixin Cao, Minghui Liu, Lei Hou, and Juanzi Li. 2021. Learning from miscellaneous other-class words for few-shot 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 6236–6247, Online. Association for Computational Linguistics.

Ralph Weischedel, Martha Palmer, Mitchell Marcus, Eduard Hovy, Sameer Pradhan, Lance Ramshaw, Nianwen Xue, Ann Taylor, Jeff Kaufman, Michelle Franchini, et al. 2013. Ontonotes release 5.0 ldc2013t19. *Linguistic Data Consortium, Philadelphia, PA*, 23.

Jinhai Yang, Hua Yang, and Lin Chen. 2021. Towards cross-granularity few-shot learning: coarse-to-fine pseudo-labeling with visual-semantic meta-embedding. In *Proceedings of the 29th ACM International Conference on Multimedia*, pages 3005–3014.

Yi Yang and Arzoo Katiyar. 2020. Simple and effective 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, Online. Association for Computational Linguistics.

Jianguo Zhang, Trung Bui, Seunghyun Yoon, Xiang Chen, Zhiwei Liu, Congying Xia, Quan Hung Tran, Walter Chang, and Philip Yu. 2021. Few-shot intent detection via contrastive pre-training and fine-tuning. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 1906–1912, Online and Punta Cana, Dominican Republic. Association for Computational Linguistics.

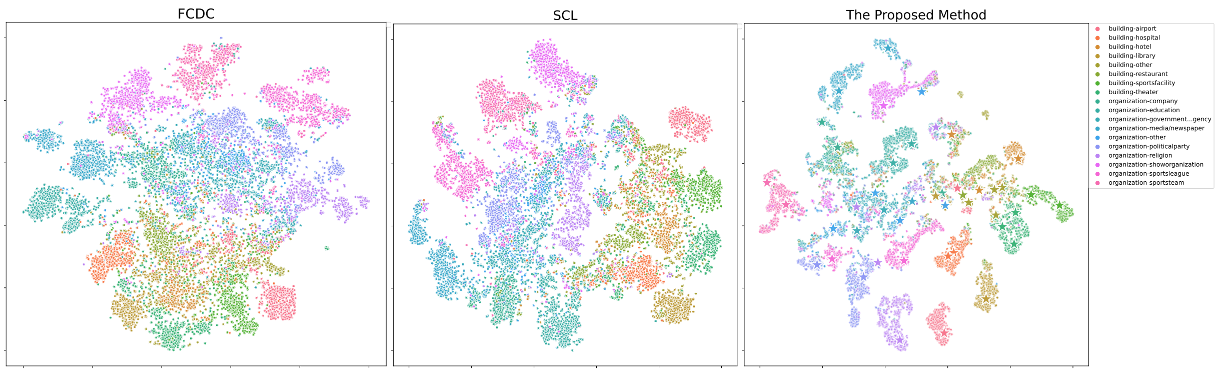

Figure 5: T-SNE visualization of **FCDC** (left), **SCL** (Medium), and **the proposed method** (right). We further annotate the retrieved prototypes ($\star$) of each fine class through Prototype Retrieval.

A Appendix

A.1 Implementation Details

For the proposed method and all baselines, we implemented based on the BERT-base-cased pre-trained model[4]. We trained the models for 3 epochs during coarse training and trained the models for 20 and 30 epochs during 5-shot and 1-shot, respectively. For our method and BERT, we set batchsize=32 and learning_rate=5e-5 on coarse training and batchsize=4 and learning_rate=1e-4 on few-shot learning. For other methods, we used their default parameters. For all methods, we choose the last model after few-shot training for evaluation. All our experiments are conducted on NVIDIA GeForce RTX 3090.

For our method, we performed clustering using the K-means clustering (Hartigan and Wong, 1979) [5] on the BERT model trained after the first epoch, and then re-initialized and trained on the coarse classes for 3 epochs. We set the cluster number of each coarse class to 50 for both in-domain and cross-domain settings. We set the margin $m = 2$ for the in-domain setting and $m = 0.2$ for the cross-domain setting. For the prototype boosting, we set $T = 3, K = 6$.

A.2 Visualization of More Baselines

In Figure 5, we show the t-SNE visualization of more competitive baselines. As we can see, all baseline models show less separable representations after coarse training due to disregard for the structure of sub-classes, while the proposed model can achieve quite fine-friendly representations.

A.3 Details of the Medium-fine dataset

In order to evaluate different methods with various fine granularity, we constructed a Medium-fine dataset (39 entity classes) based on the original Fine dataset (66 entity classes) by merging similar classes into new classes. The details of the merged classes can be found in 6. For other details of the Fine dataset, please refer to (Ding et al., 2021).

[4]https://github.com/huggingface/transformers

[5]https://scikit-learn.org/stable/modules/generated/sklearn.cluster.KMeans.html

New class name	# Entity	Included classes
art-music/film/ broadcastprogram	1826	art-music, art-film, art-broadcastprogram
art-other	416	art-other
art-writtenart/painting	814	art-writtenart,art-painting
building-airport/other	2211	building-airport,building-other
building-hospital	261	building-hospital
building-hotel/restaurant	395	building-hotel,building-restaurant
building-theater/ sportsfacility/library	901	building-theater,building-sportsfacility,building-library
event-attack/battle/war/ militaryconflict	834	event-attack/battle/war/militaryconflict
event-disaster/other	882	event-disaster,event-other
event-election/protest	268	event-election,event-protest
event-sportsevent	1140	event-sportsevent
location-GPE	15349	location-GPE
location-bodiesofwater/park	1255	location-bodiesofwater,location-park
location-island/mountain	1006	location-island,location-mountain
location-road/railway/ highway/transit/other	3009	location-road/railway/highway/transit,location-other
organization-education	1635	organization-education
organization-media/newspaper	950	organization-media/newspaper
organization-politicalparty/ government/governmentagency	1992	organization-politicalparty,organization-government/governmentagency
organization-religion/other	3782	organization-religion,organization-other
organization-showorganization/ company	3521	organization-showorganization,organization-company
organization-sportsteam/sportsleague	2571	organization-sportsteam,organization-sportsleague
other-astronomything	497	other-astronomything
other-award	710	other-award
other-chemicalthing/medical/ biologything/disease	3032	other-chemicalthing,other-medical,other-biologything,other-disease
other-currency	576	other-currency
other-educationaldegree	248	other-educationaldegree
other-god	468	other-god
other-language	632	other-language
other-law	367	other-law
other-livingthing	630	other-livingthing
person-actor/director	1573	person-actor,person-director
person-artist/author/scholar	3227	person-artist/author,person-scholar
person-athlete	2193	person-athlete
person-politician	2122	person-politician
person-soldier/other	7149	person-soldier,person-other
product-food/other	1525	product-food,product-other
product-ship/car/airplane/train	1513	product-ship,product-car,product-airplane,product-train
product-software/game	980	product-software,product-game
product-weapon	428	product-weapon

Table 6: Details of the merged classes. **# Entity** denotes the entity number in the test set.

ACL 2023 Responsible NLP Checklist

A For every submission:

☑ A1. Did you describe the limitations of your work?
7

☑ A2. Did you discuss any potential risks of your work?
7

☑ A3. Do the abstract and introduction summarize the paper's main claims?
1

☒ A4. Have you used AI writing assistants when working on this paper?
Left blank.

B ☑ Did you use or create scientific artifacts?

4

☑ B1. Did you cite the creators of artifacts you used?
4,Appendix A

☒ B2. Did you discuss the license or terms for use and / or distribution of any artifacts?
inapplicable

☑ B3. Did you discuss if your use of existing artifact(s) was consistent with their intended use, provided that it was specified? For the artifacts you create, do you specify intended use and whether that is compatible with the original access conditions (in particular, derivatives of data accessed for research purposes should not be used outside of research contexts)?
4

☒ B4. Did you discuss the steps taken to check whether the data that was collected / used contains any information that names or uniquely identifies individual people or offensive content, and the steps taken to protect / anonymize it?
The datasets we use are publicly available. And we need to perform NER tasks that involve identifying the names of people, thus the data are usually not anonymized.

☑ B5. Did you provide documentation of the artifacts, e.g., coverage of domains, languages, and linguistic phenomena, demographic groups represented, etc.?
4

☑ B6. Did you report relevant statistics like the number of examples, details of train / test / dev splits, etc. for the data that you used / created? Even for commonly-used benchmark datasets, include the number of examples in train / validation / test splits, as these provide necessary context for a reader to understand experimental results. For example, small differences in accuracy on large test sets may be significant, while on small test sets they may not be.
4, Appendix B

C ☑ Did you run computational experiments?

4

☑ C1. Did you report the number of parameters in the models used, the total computational budget (e.g., GPU hours), and computing infrastructure used?
Appendix A

The Responsible NLP Checklist used at ACL 2023 is adopted from NAACL 2022, with the addition of a question on AI writing assistance.

☑ C2. Did you discuss the experimental setup, including hyperparameter search and best-found hyperparameter values?
Appendix A

☑ C3. Did you report descriptive statistics about your results (e.g., error bars around results, summary statistics from sets of experiments), and is it transparent whether you are reporting the max, mean, etc. or just a single run?
4

☑ C4. If you used existing packages (e.g., for preprocessing, for normalization, or for evaluation), did you report the implementation, model, and parameter settings used (e.g., NLTK, Spacy, ROUGE, etc.)?
Appendix A

D ☒ Did you use human annotators (e.g., crowdworkers) or research with human participants?
Left blank.

☐ D1. Did you report the full text of instructions given to participants, including e.g., screenshots, disclaimers of any risks to participants or annotators, etc.?
No response.

☐ D2. Did you report information about how you recruited (e.g., crowdsourcing platform, students) and paid participants, and discuss if such payment is adequate given the participants' demographic (e.g., country of residence)?
No response.

☐ D3. Did you discuss whether and how consent was obtained from people whose data you're using/curating? For example, if you collected data via crowdsourcing, did your instructions to crowdworkers explain how the data would be used?
No response.

☐ D4. Was the data collection protocol approved (or determined exempt) by an ethics review board?
No response.

☐ D5. Did you report the basic demographic and geographic characteristics of the annotator population that is the source of the data?
No response.