Hybrid Enhancement-Based Prototypical Networks for Few-Shot Relation Classification

Lei Wang
Soochow University
Jianfeng Qu (jfqu@suda.edu.cn)
Soochow University
Tianyu Xu
Soochow University
Zhixu Li
Fudan University
Wei Chen
Soochow University
Jiajie Xu
Soochow University
Lei Zhao
Soochow University

Research Article

Keywords: Few-shot learning, relation classification, prototypical networks, enhancement-based, unbiased relation prototypes

Posted Date: May 26th, 2022

DOI: https://doi.org/10.21203/rs.3.rs-1684382/v1

Hybrid Enhancement-Based Prototypical Networks for Few-Shot Relation Classification

Lei Wang[1], Jianfeng Qu[1*], Tianyu Xu[1], Zhixu Li[2], Wei Chen[1], Jiajie Xu[1] and Lei Zhao[1]

[1]School of Computer Science and Technology, Soochow University, Suzhou, 215006, China.
[2]School of Computer Science, Fudan University, Shanghai, 201203, China.

*Corresponding author(s). E-mail(s): jfqu@suda.edu.cn;
Contributing authors: lwangcs@stu.suda.edu.cn;
tyxutianyu@stu.suda.edu.cn; zhixuli@fudan.edu.cn;
robertchen@suda.edu.cn; xujj@suda.edu.cn; zhaol@suda.edu.cn;

Abstract

Few-shot relation classification is to recognize the semantic relation between an entity pair with very few samples. Prototypical network has proven to be a simple yet effective few-shot learning method for relation extraction. However, under the condition of data scarcity, the relation prototypes we achieve are usually biased compared to the real ones computed from all samples within a relation class. To alleviate this issue, we propose hybrid enhancement-based prototypical networks. In particular, our model contains three main enhancement modules: 1) a query-guided prototype enhancement module using rich interactive information between the support instances and the query instance as guidance to obtain more accurate prototype representations; 2) a query enhancement module to diminish the distribution gap between the query set and the support set; 3) a support enhancement module adopting a pseudo-label strategy to expand the scale of available data. On basis of these modules, we further design a novel prototype attention fusion mechanism to fuse information and compute discriminative relation prototypes for classification. In this way, we hope to obtain unbiased representations closer to our expected prototypes by improving the available data scale and data utilization efficiency. Extensive experimental results on the widely-used FewRel dataset demonstrate the superiority of our proposed model.

Keywords: Few-shot learning, relation classification, prototypical networks, enhancement-based, unbiased relation prototypes

1 Introduction

Relation classification (RC) is a fundamental task in information extraction (IE), aiming to identify the relation between two given entities in a sentence. Traditional relation classification models often require a lot of manual annotation data which is time-consuming and labor-intensive. To deal with this issue, [1] propose the distant supervision method with the assumption that if two entities have a relation in the knowledge base (KB), then all the sentences that mention these two entities in the corpus will be regarded as training instances of this relation. However, when faced with the long-tail distribution in the real scene [2], the distant supervision method usually fails to quickly generalize to novel relations due to its inner mechanism of requiring abundant training data.

[3] first formulate RC as a few-shot learning task with only a handful of training instances, shown in Table 1. Since its practical significance, many efforts have been devoted to this scenario. For instance, meta-learning, also known as "learning to learn", learns model initialization that fast adapts to new tasks through learning on the meta-train tasks [4–6]. Further, a simple yet effective meta-learning based few-shot learning method is prototypical network [7]. The main idea of prototypical network is that there exists a prototype for each class so that we can learn the representation of the prototype in the embedding space and finally classify all the query instances via the nearest neighbor rule. Specifically, [8] propose a multi-level matching and aggregation prototypical network so that each query and support's matching information can be utilized. [9] enhance prototypical networks with instance-level and feature-level attention schemes to focus on crucial instances and features respectively. [10] adopt contrastive learning and task-adaptive training strategy to pay more attention to hard tasks.

Despite their remarkable performance, they often ignore that the obtained prototype has an extreme bias towards the limited given support set. As illustrated in Fig. 1, our expected prototype, representing the center of the whole relation class, should be computed by the mean value of all samples within a class. Unfortunately, under the few-shot scenario, only a few training samples are available from the support set, and then the prototype (i.e., the triangle in Fig. 1) computed from it seriously deviates from the expected one (i.e., the pentagram). Consequently, the obtained biased prototype becomes an inconsiderate metric for classifying these query instances, degrading the performance of the prototypical network.

On the other hand, previous works tend to treat the support set and query set as two separate parts, ignoring rich interactive information between the support instances and the query instances. That is, traditional prototypes are computed by averaging all the instances in the support set wherever the query

Table 1 An easy example for a 3-way 1-shot scenario. The red and blue words correspond to the head entity and tail entity respectively.

Support Set	
class A: date_of_birth	Mark Twain was born in 1835.
class B: capital_of	London is the capital of the U.K.
class C: founder	Microsoft was founded by Bill Gates.
Query Instance	
class A or B or C	Washington is the capital of the U.S.A.

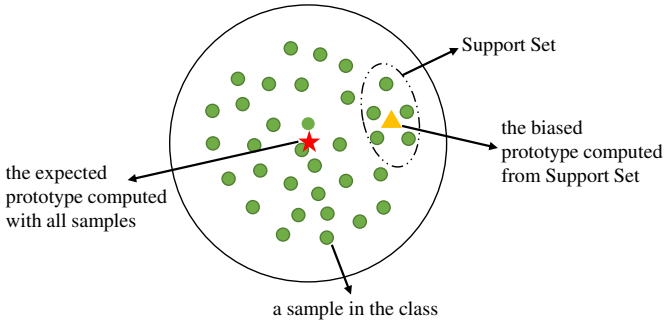

Fig. 1 An example shows the deviation between the biased prototype computed from the support set and the expected prototype computed from all samples.

instance locates. Nevertheless, we sometimes encounter points that are far from the center (near the class boundary) but still fall into this category. In this case, a single unified prototype might be inappropriate and classify the query instance by mistake. Moreover, due to the limitation of available data, there often exists a distribution gap between the support set and query set. If ignoring the gap and directly using the original query instances, we will get a problematic distance from the query to the prototype brought by the underlying gap during the predicting process.

To alleviate these problems, we propose **H**ybrid **E**nhancement-based **P**rototypical **N**etworks (HEPN). Our goal is to obtain accurate and unbiased prototype representations as possible. According to the above analysis, we summarize that the available data scale and data utilization efficiency are two key factors restricting the representational ability of class prototypes in few-shot learning. To this end, our model consists of three enhancement modules. In particular, to address the limited data scale in the support set, a support enhancement module is introduced to expand the available data. Our method is based on a naive proposition: the more samples supplied, the more accurate the prototype representations obtained. Although we possess only K (1 or 5) support samples, there are lots of unlabeled query samples. So, we adopt a pseudo-label method to utilize these unlabeled but informative query samples. With pseudo-labeled query samples added to the support set, more

support instances are available for computing more accurate and unbiased relation prototypes. As for data utilization efficiency, a query-guided prototype enhancement module and a query enhancement module are introduced. With regard to the query-guided prototype enhancement module, our objective is to compute query-specific prototypes. When given a query, our module will capture interactive information between the support instances and the query instance as guidance to assign reasonable weights to each support instance. Besides, considering the distribution gap between the support set and query set, we propose a query enhancement module to diminish the gap. By adding a shifting term to the query samples, the query set will distribute more closely to the support set. Utilizing these enhanced query samples can reflect more real distance from these queries to each prototype.

With the above three enhancement modules, we are able to promote both available data scale and data utilization efficiency without using extra data. In order to further effectively fuse the information of basic prototypes and enhanced prototypes and reduce the impact of misclassified samples, we develop a novel prototype attention fusion module that can improve final classification performance. The extensive experimental results on the FewRel dataset demonstrate that our model can produce more accurate prototypes, and achieve higher accuracy on downstream tasks. In conclusion, our main contributions are summarized as follows:

- We investigate two key factors restricting the representational ability of class prototypes in few-shot learning RC: the data scale and data utilization efficiency.
- We propose a hybrid enhancement-based prototypical network that is conceptually simple but effective to generate more accurate prototypes.
- To enlarge the data scale without extra human annotations, we design a support enhancement module to increase the number of available training samples with the help of confidently predicted query instances.
- To promote data utilization efficiency, we exploit the rich interactive information between the support set and the query set and propose a query-guided prototype enhancement module and a query enhancement module. Moreover, a novel prototype attention fusion module is employed to further combine the useful information from basic and enhanced prototypes.
- Extensive experiments on the FewRel dataset verify the superiority of our proposed model compared with various strong baselines.

The rest of this paper is organized as follows. We formulate the few-shot relation classification problem in Section 2. Section 3 describes our HEPN model in detail. We conduct experiments on the FewRel dataset to demonstrate the effectiveness of our model in Section 4. The related work is presented in Section 5, which is followed by the conclusion and the future work in Section 6.

2 Task Definition

In few-shot RC, we are given a dataset $D_{meta-test}$, which is split into two parts: $D_{test-support}$ and $D_{test-query}$. There are N relation classes in the $D_{test-support}$, each relation class has K labeled samples, and that's why it is called N-way K-shot task. The $D_{test-query}$ contains the same N classes as the $D_{test-support}$, and our goal is to classify these unlabeled query samples in the $D_{test-query}$ through the knowledge learned in the $D_{test-support}$. As the K is usually very small (1 or 5), it's difficult to train a good model from scratch. So usually we are given an auxiliary dataset $D_{meta-train}$, which contains a lot of labeled data to help train the model. Note that the classes in the $D_{meta-train}$ and $D_{meta-test}$ are disjoint with each other. In other words, query samples in the $D_{test-query}$ are novel relations that cannot be seen at the training stage.

One popular approach is the paradigm proposed by [11], which ensures that train and test conditions must match. Specifically, in each training iteration, also known as the episode, we randomly select N classes from $D_{meta-train}$, and in each class, K support instances are selected. In this way, we construct a support set $S = \{s_k^i;\ i = 1,...,N, k = 1,...,K\}$, where s_k^i is the k-th instance of class i. Meanwhile, R instances are randomly selected from the remaining samples of the N classes to construct a query set $Q = \{q_j;\ j = 1,...,R\}$. Each instance consists of a set of samples (x,p,r), where x is a sentence, $p = (p_1, p_2)$ are the positions of the head entity and tail entity, and r is the relation label.

At the training stage, our goal is to optimize the following objective function:

$$L = -\frac{1}{R} \sum_{(q,y) \in Q} p(y|S,q) \tag{1}$$

and $p(y \mid S,q)$ is computed as follows:

$$p(y \mid S,q) = \frac{exp(-d(q,P_i))}{\sum_{j=1}^{N} exp(-d(q,P_j))} \tag{2}$$

where P_i is the prototype of class i, and $\mathrm{d}(\cdot,\cdot)$ is the Euclidean distance function. The focus of this paper is on how to obtain accurate and unbiased prototypes with only a handful of samples.

3 Methodology

In this section, we will introduce our proposed hybrid enhancement-based prototypical network (HEPN) in detail. The framework of the HEPN is shown in Fig. 2, which mainly consists of six components:

- **Sentence Encoder**. Given an instance with the positions of the head entity and tail entity, the pre-trained language model BERT [12] is employed to get the contextualized representation of the sentence.
- **Query-Guided Prototype Enhancement Module**. Given the representation of each instance in the support set and query set, the query-guided

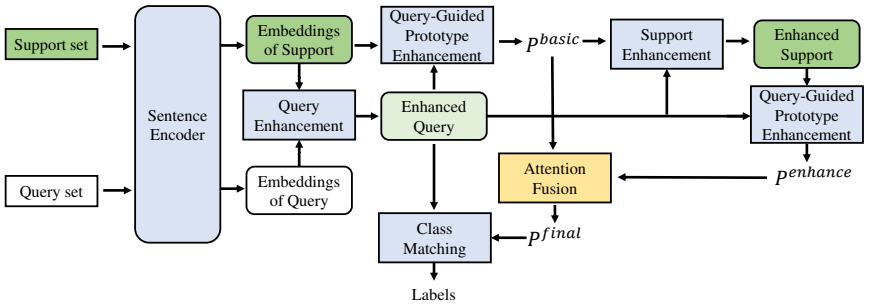

Fig. 2 The framework of our proposed HEPN model.

prototype enhancement module computes query-specific prototypes for each query. In other words, different relation prototypes will be computed based on the semantic relevance between the support set and the query instance.

- **Query Enhancement Module**. Given the contextualized embeddings of the support instances and query instances, the query enhancement module computes the enhanced query instances which distribute more closely to the support set by adding a shifting term to each query. This module will merely be adopted at the inference stage.
- **Support Enhancement Module**. Given the basic prototypes and the enhanced query instances, the support enhancement module adopts a pseudo-label strategy to classify these unlabeled query samples into one of the N relations. After picking out high-confidence pseudo-labeled query samples and adding them to the support set, we get the enhanced support set. With enhanced support set, we employ our query-guided prototype enhancement module again to compute the enhanced prototype which is closer to our expected prototype. Note that the support enhancement module will only be employed at the inference stage the same as the query enhancement module.
- **Prototype Attention Fusion Module**. Given the basic prototypes and the enhanced prototypes, the prototype attention fusion module calculates the final prototypes by fusing their information and reducing the weight of noise samples. Then the final prototypes will be used for classification in the class matching module.
- **Class Matching**. Given the final prototype representations for each relation class and the enhanced query instances, the class matching module calculates the matching scores between each enhanced query and each relation class. If a relation class gets the highest matching score, our model will choose it as the final prediction.

At the training stage, only the query-guided prototype enhancement module will be employed to train a good sentence encoder. At the inference stage, we activate the query enhancement module, support enhancement module, and prototype attention fusion module to improve prediction performance without

introducing extra parameters that need to be trained. The process for few-shot relation classification of our HEPN model at the inference stage is illustrated by Algorithm 1. More details of our proposed modules will be introduced in the following subsections.

Algorithm 1 Architecture of HEPN at the inference stage

Input: Support set $S = \{s_k^i;\ i = 1,...,N, k = 1,...,K\}$
Query set $Q = \{q_j;\ j = 1,...,R\}$
Output: Prediction results $L = \{label_j;\ j = 1,...,R\}$

1: Utilize the sentence encoder BERT to get each instance's contextualized representations in S and Q with Eq.(3)
2: Update each query's representation with Eq.(7),(8): $q \rightarrow q^{enhance}$
3: Utilize the query-guided prototype enhancement module to compute basic prototypes P^{basic} with Eq.(5),(6)
4: Utilize a pseudo-label strategy to select out top Z confidently predicted query instances per relation Q^Z_{pseudo}
5: Update Support set S with Q^Z_{pseudo}: $S \rightarrow S^{enhance} = S \cup Q^Z_{pseudo}$
6: Utilize the query-guided prototype enhancement module to compute enhanced prototypes $P^{enhance}$ with Eq.(9),(10)
7: Utilize the prototype attention fusion module to compute final prototypes P^{final} with Eq.(11),(12),(13)
8: Compute the matching scores between each enhanced query $q^{enhance}$ and each relation prototype P^{final} with Eq.(14)
9: Assign each query to the relation class with the highest matching score with Eq.(15): $label_j \rightarrow q_j$

3.1 Sentence Encoder

Each instance can be represented by a sentence with the positions of the head entity and tail entity. Our sentence encoder module is adopted to encode the sentence into the embedding space. The pre-trained language model has been shown to be effective in many NLP tasks. Following the recent trend, we employ BERT [12] as the sentence encoder to get contextualized representations of the support and query instances.

Given a sentence $x = \{w_1,w_2,...,w_n\}$ in the support set or query set, where $w_i \in x$ is the word token in the sentence, the sequences $\{w_{h_s},...,w_{h_e}\}$ and $\{w_{t_s},...,w_{t_e}\}$ represent head entity mentions e_h and tail entity mentions e_t respectively, we first need to construct it in the form: {[CLS], w_1,...,[E_h],w_{h_s},...,w_{h_e},[/E_h],...,[E_t],w_{t_s},...,w_{t_e},[/E_t],...,w_n, [SEP]} to match the input of BERT. Then a special [PAD] token is used to pad the sentence to the maximum length we set.

By feeding the input to the BERT model, we can obtain the instance's contextual embedding $\mathbf{E} = \{h_0, h_1, ..., h_n, h_{n+1}\}$, where $\mathbf{E} \in \mathbb{R}^{d_{(n+2)} \times d_b}$, d_b is

the BERT hidden size, h_0 is the [CLS] token embedding, h_{n+1} is the [SEP] token embedding, and $h_i, i \in [1, n]$ is the embedding of each token. Note that n is different from the input sentence length now. Following the work of [13], we concatenate the hidden states corresponding to the start tokens of two entity mentions as the instance's representation. Specifically, the instance's representation x_i can be denoted as:

$$x_i = h^i_{E_h} \bigoplus h^i_{E_t} \tag{3}$$

where $\bigoplus$ is the concatenation operation.

3.2 Query-Guided Prototype Enhancement Module

Conventionally, the prototype P_n of class n is computed as follows:

$$P_n = \frac{1}{K} \sum_{i=1}^{K} s^n_i \tag{4}$$

where s^n_i is the i-th support instance of class n. However, simply averaging all the support samples as the class prototype regardless of which query is given will lose rich interaction between the support instances and the query instances. Although using the attention mechanism changes the assignment of the weights, it does not inherently solve the problem.

In order to enhance the connection between the support instances and the query instances, we propose a query-guided prototype enhancement module. Our starting point is to obtain query-specific prototypes for each given query. Obviously, it is unreasonable to classify all the query instances in an episode with common prototypes. Because some singular points far from the center are still belong to this class. So we want to utilize the rich interactive information between the support instances and the query instances as guidance to help compute the customized representations of relation prototypes for each query. In this way, our prototype is able to store some preliminary knowledge about the query which is beneficial for subsequent classification. Our query-specific prototypes are calculated based on the semantic relevance between the support set and the query. Specifically, the query-guided prototype can be represented as follows:

$$P^{basic}_n = \sum_{i=1}^{K} w^n_i \cdot s^n_i \tag{5}$$

where w^n_i is the weight indicating the semantic relevance between the support instance and the query instance. The weight w^n_i is computed as follows:

$$w^n_i = \frac{exp(-d(s^n_i, q))}{\sum_{j=1}^{K} exp(-d(s^n_j, q))} \tag{6}$$

where d$(\cdot,\cdot)$ is the Euclidean distance function.

With the query-guided prototype enhancement module, our model will compute distinctive prototypes for each given query which is more reasonable than previous works. At the same time, taking the semantic relevance between the support set and the query into consideration improves data utilization efficiency.

3.3 Query Enhancement Module

In the domain adaptation problem, one typical approach is to minimize the domain gap. Motivated by this, we design a query enhancement module to minimize the gap between the support set and query set. Considering that the support set and the query set are supposed to distribute in the same domain, so the gap we want to minimize refers to the distribution gap between them rather than the domain gap. Following [14], we use the mean value of the support set and the query set to represent them respectively. So the distribution gap ξ between the support set and the query set is defined as follows:

$$\xi = \frac{1}{|S|}\sum_{i=1}^{|S|} s_i - \frac{1}{|Q|}\sum_{j=1}^{|Q|} q_j \tag{7}$$

where s_i is the i-th instance in the support set and q_j is the j-th instance in the query set. Then the ξ is added to each query to get the enhanced query:

$$q^{enhance} = q + \xi \tag{8}$$

By adding a shifting term ξ, the query set will distribute more closely to the support set, which may help to classify the query with the prototypes computed by the support set. From the point of view of data utilization, our query enhancement module doesn't treat the support set and query set as two separate parts and incorporates the support set's distribution information in the query set.

3.4 Support Enhancement Module

In the few-shot scenario, only a handful of training samples are available. With only K instances per relation, it's difficult for us to obtain an accurate prototype representation. To deal with this issue, we propose a support enhancement module that can increase the number of available support samples without using extra data.

Though available support samples are very limited, we still have lots of unlabeled query samples. Inspired by [14], we adopt a pseudo-labeling strategy to make use of these unlabeled data, which assigns pseudo labels to the unlabeled query samples according to their prediction confidence. Specifically, in order to get the prediction scores, we first use our basic prototypes computed in the previous section to classify unlabeled query instances which have been enhanced by our query enhancement module. The higher the score, the

more confidence the model has in the predicted result. Next, top Z confidently predicted query instances per relation are selected with their pseudo labels to enhance the support set S. Now we have an enhanced support set with Z confidently predicted query instances: $S^{enhance} = S \cup Q^{Z}_{pseudo}$.

Since some query samples may be misclassified, simply averaging all the samples in the enhanced support set with the same weights is likely to lead to error accumulation. In order to reduce the weight of the potential misclassified sample, we adopt our query-guided prototype enhancement module again to obtain the enhanced prototype representation:

$$P_n^{enhance} = \sum_{i=1}^{K+Z} w_i^n \cdot x_i^n \tag{9}$$

where x_i^n is the i-th instance in the enhanced support set of class n. The weight w_i^n is computed as follows:

$$w_i^n = \frac{exp(-d(x_i^n, q^{enhance}))}{\sum_{j=1}^{K+Z} exp(-d(x_j^n, q^{enhance}))} \tag{10}$$

where $\mathrm{d}(\cdot,\cdot)$ is the Euclidean distance function.

Adopting our query-guided prototype enhancement module in this step can not only reduce the weight of samples that may be misclassified but also take the interactive information into consideration so as to further gain performance. The enhanced prototype we compute with more available samples is closer to our expected prototype which is supposed to be computed by averaging all the samples within a class.

3.5 Prototype Attention Fusion Module

Though we have employed our query-guided prototype enhancement module to reduce the impact of noise samples in the previous section, we can further reduce the weights of misclassified samples. To this end, we design a novel prototype attention fusion module that can fuse the information of basic prototypes and enhanced prototypes. Our starting point is to measure the important degree of basic prototypes and enhanced prototypes. And we will assign different weights to basic prototypes and enhanced prototypes according to the semantic relevance between the query and the prototypes. Specifically, the final prototype representation can be calculated as follows:

$$P_n^{final} = w_b \cdot P_n^{basic} + w_e \cdot P_n^{enhance} \tag{11}$$

where w_b and w_e are scale weight values and $w_b + w_e = 1.0$. The weight w_b and w_e can be computed as follows:

$$w_b = \frac{exp(-d(q^{enhance}, P_n^{basic}))}{exp(-d(q^{enhance}, P_n^{basic})) + exp(-d(q^{enhance}, P_n^{enhance}))} \tag{12}$$

$$w_e = \frac{exp(-d(q^{enhance}, P_n^{enhance}))}{exp(-d(q^{enhance}, P_n^{basic})) + exp(-d(q^{enhance}, P_n^{enhance}))} \tag{13}$$

where d(·,·) is the Euclidean distance function. With the prototype attention fusion module, our computed final prototypes can fuse the information between the original support set and the enhanced support set as well as those query instances. All this information complements each other and finally, we calculate accurate and unbiased prototypes that we need.

3.6 Class Matching

At the training stage, we merely adopt our query-guided prototype enhancement module to compute the basic prototypes and the class matching function based on the nearest neighbor rule is shown as Equation 2. At the inference stage, we have obtained the final prototypes which are more accurate through three enhancement modules and a prototype attention fusion module. Now we can update the equation to improve the prediction accuracy. The probability of predicting query q belongs to class i is computed as follows:

$$p(y = i|S, q) = \frac{exp(-d(q^{enhance}, P_i^{final}))}{\sum_{j=1}^{N} exp(-d(q^{enhance}, P_j^{final}))} \tag{14}$$

where d(·,·) is the Euclidean distance function. Then a query q is assigned to the class i with the highest matching score:

$$label = \arg\max_i p(y = i|S, q) \tag{15}$$

4 Experiments

4.1 Datasets and Evaluation

We evaluate our model on FewRel[1] [3], which was first generated by distant supervision and then annotated by crowd workers. It has 64 relations for training, 16 relations for validation, 20 relations for test, and each has 700 instances. We evaluate our HEPN in four few-shot learning configurations: 5-way 1-shot, 5-way 5-shot, 10-way 1-shot, and 10-way 5-shot. We evaluate our model with 10,000 random sampled tasks from the validation set and compute the average accuracy according to the official evaluation scripts. Note that in order

[1] https://www.zhuhao.me/fewrel/

Table 2 Hyper-parameters of our approach.

Component	Parameter	Value
BERT	type	base-uncased
	hidden size	768
	max length	128
Dataset	number of query per class	15
Training	learning rate	2e-5
	max iterations	20000
	dropout rate	0.1
Inference	number Z for 5-way 1-shot	15
	number Z for 5-way 5-shot	14
	number Z for 10-way 1-shot	15
	number Z for 10-way 5-shot	13

Table 3 Accuracy(%) of few-shot classification on the FewRel validation set with different numbers of pseudo-labeled samples. The **best** numbers are highlighted in each column.

Number Z	5-way 1-shot	5-way 5-shot	10-way 1-shot	10-way 5-shot
12	90.35	95.42	81.11	92.04
13	90.47	95.92	81.45	**93.04**
14	91.33	**96.43**	81.61	92.85
15	**91.40**	96.41	**82.47**	92.04
16	91.24	96.08	80.80	91.63

to preserve the fairness of test results, the test set doesn't release to the public. Most of our experiments and performance analysis are conducted on the validation set of FewRel, and we only report the final official results on the nonpublic test set.

4.2 Implementation Details

All the hyper-parameters we use are shown in Table 2. We implement our approach with Pytorch [15]. We initialize the word embeddings with the bert-base-uncased model [12]. The max sentence length is set to be 128, and the word embedding dimension is 768. We employ Adam as our optimizer to minimize the loss and the initial learning rate is set to be 2e-5. Our model is trained on 4 NVIDIA GeForce GTX 1080 Ti GPUs. It is worth noting that, due to the insufficient computing power of our GPU, we use a single episode per batch. We train 20,000 iterations and the model is evaluated at every 2000 iterations. The dropout rate of 0.1 is used to avoid overfitting. Each episode contains 15 query instances per class. As for the number of pseudo-labeled samples Z, we perform grid search to find the best parameter setting. As the experimental results shown in Table 3, we can see that $Z = 15$ can obtain the best evaluation accuracy on 5-way 1-shot and 10-way 1-shot tasks, and on 5-way 5-shot and 10-way 5-shot tasks, the best evaluation accuracy can be obtained when $Z = 14$ and $Z = 13$ respectively.

Table 4 Accuracy(%) of few-shot classification on the FewRel validation / test set. The **best** and second-best numbers are highlighted in each column.

Model	5-way 1-shot	5-way 5-shot	10-way 1-shot	10-way 5-shot	Average
Proto-CNN[7]	72.65 / 74.52	86.15 / 88.40	60.13 / 62.38	76.20 / 80.45	73.78 / 76.44
Proto-BERT[7]	82.92 / 80.68	91.32 / 89.60	73.24 / 71.48	83.68 / 82.89	82.79 / 81.16
Proto-HATT[9]	75.01 / — —	87.09 / 90.12	62.48 / — —	77.50 / 83.05	75.52 / — —
MLMAN[8]	79.01 / 82.98	88.86 / 92.66	67.37 / 75.59	80.07 / 87.29	78.83 / 84.63
BERT-PAIR[16]	85.66 / 88.32	89.48 / 93.22	76.84 / 80.63	81.76 / 87.02	83.44 / 87.30
TD-Proto[17]	— — / 84.76	— — / 92.38	— — / 74.32	— — / 85.92	— — / 84.35
CTEG[18]	84.72 / 88.11	92.52 / 95.25	76.01 / 81.29	84.89 / 91.33	84.54 / 89.00
HCRP[10]	90.90 / 93.76	93.22 / 95.66	**84.11** / **89.95**	87.79 / 92.10	89.01 / 92.87
CPSE[19]	87.21 / 90.40	94.86 / 96.95	80.34 / 84.68	91.36 / **94.15**	88.44 / 91.55
IAN[20]	— — / 90.77	— — / 96.18	— — / 84.65	— — / 93.15	— — / 91.19
HEPN(ours)	**91.40** / **94.93**	**96.43** / **97.17**	82.47 / 88.78	**93.04** / 93.78	**90.84** / **93.67**

4.3 Baselines

We compare our proposed model with the following strong baselines:

- **Proto**[7], the algorithm of prototypical networks. CNN and BERT are employed as the encoder separately (**Proto-CNN** and **Proto-BERT**).
- **Proto-HATT**[9], prototypical networks with instance-level and feature-level attention schemes to focus on crucial instances and features.
- **MLMAN**[8], a multi-level matching and aggregation prototypical network to take each query and support's matching information into consideration.
- **BERT-PAIR**[16], a method measuring the similarity of sentence pairs.
- **TD-Proto**[17], a method using relation and entity descriptions to enhance the prototypical network.
- **CTEG**[18], a prototypical network-based method with entity-guided attention and confusion-aware training to deal with the relation confusion problem.
- **HCRP**[10], a prototypical network-based method with contrastive learning and task-adaptive training strategy to pay more attention to hard tasks.
- **CPSE**[19], a two-stage approach with supervised contrastive learning in the pre-training stage and a sentence- and entity-level prototypical network in the meta-learning stage to improve the performance.
- **IAN**[20], an interactive attention network using inter-instance and intra-instance interactive information to classify the relations.

4.4 Overall Evaluation Results

Table 4 presents the experimental results on the FewRel validation set and test set. Compared to the competitive baseline models, our proposed method achieves the best results on 5-way 1-shot and 5-way 5-shot tasks, comparable results on 10-way 1-shot and 10-way 5-shot tasks, and the best average performance on four tasks. Specifically, our method achieves 94.93% test accuracy on 5-way 1-shot task, 97.17% test accuracy on 5-way 5-shot task, and 93.67% average test accuracy, which are 1.17%, 0.22%, and 0.80% improvement

Table 5 Experimental results of the ablation studies. All the results reported in this section are on the validation set of FewRel.

Model	5-way 1-shot	5-way 5-shot	10-way 1-shot	10-way 5-shot	Average
entire HEPN	91.40	96.43	82.47	93.04	90.84
w/o query-guided prototype	89.85	95.81	79.96	91.99	89.40(↓**1.44**)
w/o query enhancement	90.90	96.15	82.19	92.79	90.51(↓**0.33**)
w/o support enhancement	88.67	95.94	78.43	92.14	88.80(↓**2.04**)
w/o attention fusion	91.33	95.95	82.27	92.53	90.52(↓**0.32**)

respectively compared with the current state-of-the-art baselines, demonstrating the effectiveness of our method. On the 10-way 5-shot task, although our model only achieves comparable performance on the test set, our model obtains a 1.68% improvement on the validation set compared with the suboptimal results. On the 10-way 1-shot task which is the most difficult task among these four tasks, our HEPN is slightly lower than HCRP which introduces external relation label name and description information. We think that the lower performance on the 10-way 1-shot task is due to the fact that the additional relation label name and description information that HCRP uses will benefit the instance representation especially when there is only one support instance, while our method does not introduce any external information.

The performance gain of our model mainly comes from four aspects:

- We adopt a query-guided prototype enhancement module to obtain query-specific prototypes with rich interactive information between the support instances and the query instances taken into consideration.
- We adopt a query enhancement module to diminish the distribution gap between the query set and the support set.
- Our support enhancement module employs a pseudo-label strategy to expand the support set.
- We also design a novel prototype attention fusion module to fuse information and further gain performance.

4.5 Ablation Study

In order to evaluate the effectiveness of each component in our HEPN, ablation studies are carried out. We first remove our query-guided prototype enhancement module to verify the effectiveness of query-specific prototypes with rich interactive information between the support instances and the query instances. Then we remove the query enhancement module and support enhancement module, in turn, to verify the effectiveness of the support set's distribution information and more available pseudo-labeled samples respectively. Finally, we remove the prototype attention fusion module to verify the effectiveness of information fusion. Our ablation results are shown in Table 5 and all the results are obtained on the FewRel validation set.

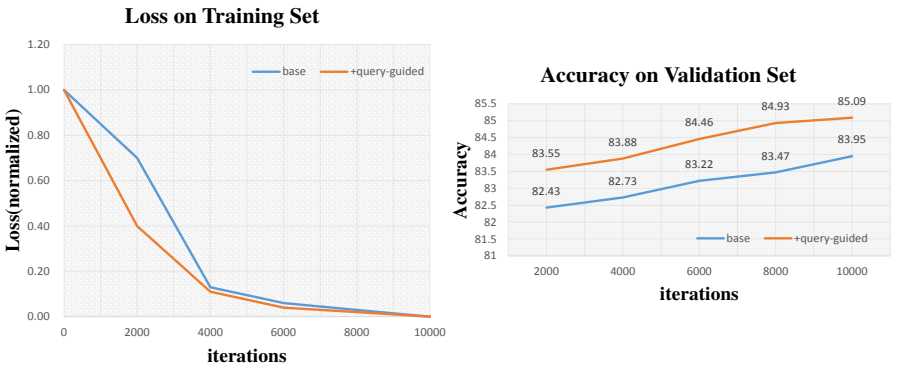

Fig. 3 The training convergence on 5-way 1-shot task using different models (base model and with query-guided prototype enhancement module).

4.5.1 Ablation of Query-Guided Prototype Enhancement Module

We first remove the query-guided prototype enhancement module, i.e., directly averaging all the representations of the instances in the support set as the prototype representation. From Table 5, we can find that each experimental configuration declined to some extent with an average decrease of 1.44%. This provides evidence that our query-guided prototype enhancement module is effective for improving performance.

In order to more clearly demonstrate the effectiveness of our query-guided prototype enhancement module, further experiments are carried out as follows: we first remove all the modules with only the sentence encoder BERT reserved as the base model, next we equip the base model with our query-guided prototype enhancement module, then we train both models 10,000 iterations on 5-way 1-shot task separately and record the training loss and accuracy on the FewRel validation set every 2000 iterations. Each accuracy record is evaluated with 6000 random sampled tasks from the validation set. The experimental results are shown in Fig. 3.

From the results, we can see that our query-guided prototype enhancement module can not only gain performance, but also speed up the convergence as the training loss of our model decreases faster than that of the base model, and the validation accuracy increases faster. This once again verifies the necessity of considering the interactive information between the support set and query set. We argue that the singular points will reduce the effectiveness of the training process because the conventional single prototype mechanism is unable to capture the actual category of these singular points. In contrast, our query-guided prototypes can excellently resolve this problem.

4.5.2 Ablation of Query Enhancement Module

We remove the query enhancement module, i.e., directly using the original query embeddings without extra operations. As shown in Table 5, our

results have an average decrease of 0.33%. The improvements demonstrate the effectiveness of the query enhancement module.

Essentially, our query enhancement module equips the query instances with the support set's distribution information. As we know, there is a distribution gap between the support set and query set due to the limitation of available data. By adding a shifting term ξ, our query instances can distribute more closely to the distribution center of the support set. That is to say, the query samples are shifted towards the whole support set. As a result, we can shorten the distance between the query set and the support set.

4.5.3 Ablation of Support Enhancement Module

We next remove the support enhancement module, i.e., directly utilizing the original support set to compute prototypes without pseudo-labeled samples. The results of all configurations have decreased to some extent no doubt with an average decrease of 2.04% which demonstrates the effectiveness of our support enhancement module. Specifically, the accuracy on the 5-way 1-shot task decreases from 91.40% to 88.67% with a decrease of 2.73%, and the accuracy on the 5-way 5-shot, 10-way 1-shot, 10-way 5-shot tasks respectively have a decrease of 0.49%, 4.04%, 0.9%. From the comparison of the decrease, we can find that our support enhancement module shows greater improvement on 1-shot tasks and a smaller boost on 5-shot tasks. This discovery may be explained by the following reason.

As we have emphasized, our expected prototypes are supposed to be computed from the mean value of all samples within a relation class. However, we only possess 1 or 5 samples in the few-shot scenario, so it's inevitable to compute biased prototypes. And data scarcity is even worse on 1-shot tasks. The unique data may be very different from the real prototype and on 5-shot tasks, this situation will be slightly better. So naturally, the pseudo-labeled samples we add to the support set make a more significant improvement for 1-shot tasks.

As for the selection of the number Z, we can find a trend from Table 3 that accuracy raises with more pseudo-labeled samples. This matches our intuition that with more available samples, more accurate prototype representations can be obtained. At the same time, we find that the results will decrease after Z reaches a certain value. We believe this is the bad influence caused by the misclassified noise samples. It is also a direction for our future work on how to further reduce the impact of noise samples.

4.5.4 Ablation of Prototype Attention Fusion Module

Finally, we remove the prototype attention fusion module, i.e., directly using the enhanced prototypes for final classification. The presented results in Table 5 show an average decrease of 0.32%. The improvements demonstrate the effectiveness of our prototype attention fusion module.

When removing the prototype attention fusion module, the number of pseudo-labeled samples Z achieving the best results has changed which is

Table 6 The change of pseudo-labeled samples' number Z when the prototype attention fusion module is removed.

Configuration	HEPN w/o attention fusion	HEPN
5-way 1-shot	13	15
5-way 5-shot	13	14
10-way 1-shot	12	15
10-way 5-shot	11	13

shown in Table 6. As we can see, our HEPN with prototype attention fusion module tends to use more pseudo-labeled samples. This may be explained by the following reason: without the prototype attention fusion module, our HEPN is likely to be influenced by the misclassified query samples which can bring a drop in results. However, our prototype attention fusion module is designed to reduce the impact of misclassified samples which will be assigned lower weights. As a result, our prototype attention fusion module can utilize more pseudo-labeled samples which can bring a performance boost.

5 Related Work

5.1 Few-shot Learning

Few-shot learning is dedicated to solving the problem of insufficient training samples which can be divided into two categories: gradient-based methods and metric-based methods.

5.1.1 Gradient-based Methods

Gradient-based methods can be considered meta-learning in a narrow sense. Parameters of gradient-based methods are optimized over tasks that correspond to different learning problems rather than samples. Specifically, there is a meta-learner and a base-learner in gradient-based methods. The meta-learner in the outer loop tries to learn the global information and initialize the base-learner while the base-learner in the inner loop aims to fine-tune the parameters and fast adapt to novel tasks with only several steps of gradient descent. [4] propose a model-agnostic meta-learning algorithm (MAML) that can train the model to be easy to fine-tune. [6] introduce meta-information to improve the performance of meta-learning which imitates the process of human learning.

5.1.2 Metric-based Methods

Metric-based methods try to learn a pairwise similarity metric in the embedding space where similar samples are closer together while relatively unlike samples are farther apart. Normally, metric-based methods contain a feature extractor and a metric function. The feature extractor is used to encode the samples into the embedding space and the metric function is adopted to compute the similarity between different samples. Finally, all the unlabeled samples

can be classified based on the nearest neighbor rule. Prototypical networks [7], Matching networks [11] and Relation Network [21] are the most representative works of metric-based methods. As prototypical networks have proven to be a simple yet effective method for few-shot relation classification, we choose it as our backbone in this paper.

5.2 Few-shot Relation Classification

Few-shot relation classification aims at recognizing two given entities' semantic relation in a sentence with only a handful of training instances. [3] first formulate relation classification as a few-shot learning task and present a large-scale benchmark FewRel for few-shot relation classification. [22] propose a large-margin prototypical network with fine-grained features to generalize on long-tail relations. [8] take each query and support's matching information into consideration in an interactive way. [9] design instance-level and feature-level attention schemes to pay more attention to crucial instances and features. [23] propose a two-phase prototypical network with prototype attention alignment and triplet loss to avoid catastrophic forgetting. [17] enhance prototypical network with relation and entity descriptions. [24] improve typical meta-learning framework with task enrichment and support classifier modules. [25] use entity concept as external information to improve performance. [10] adopt contrastive learning and task-adaptive training strategy to focus on hard tasks. [20] propose an interactive attention network that uses inter-instance and intra-instance interactive information to produce discriminative instance representations. However, the task of achieving unbiased prototypes is still under-explored. In this paper, we focus on how to improve data scale and data utilization efficiency and propose three main enhancement modules with a novel prototype attention fusion module so as to achieve unbiased prototype representations.

5.3 Semi-supervised Few-shot Learning

Semi-supervised few-shot learning usually needs extra unlabeled samples to improve performance [14, 26, 27]. [27] improve prototypical network with unlabeled samples to calculate prototypes by Soft k-Means. [28] propose a transductive propagation network (TPN) which utilizes a graph construction module to propagate labels from the labeled support set to the unlabeled query set and updates all parameters end-to-end. [26] propose a learning to self-train (LST) method which uses pseudo-labeling on unlabeled data to augment the support set, and then re-trains and fine-tunes the model. [14] employ pseudo-labeling strategy and feature shifting in order to diminish the intra-class bias and the cross-class bias respectively. Our work also adopts a pseudo-labeling strategy to expand the support set which is inspired by [14].

6 Conclusion and Future Work

In this paper, we propose a novel model HEPN to address the problem of few-shot relation classification. In order to improve the data scale of available training samples and data utilization efficiency, we propose three enhancement modules, including a query-guided prototype enhancement module, a query enhancement module and a support enhancement module, without introducing any extra data or external information. Additionally, we craft a novel prototype fusion module to combine valuable information from different aspects. In this way, our model gets more accurate representations for prototypes and queries, promoting the performance of prototypical networks. The experimental results on the FewRel dataset demonstrate the superiority of the proposed model and the effectiveness of each component. In the future, we will introduce contrastive self-supervised learning and some external knowledge like relation label name and description information to learn better informative and discriminative prototype representations.

Declarations

Ethical Approval and Consent to participate

Not applicable

Human and Animal Ethics

Not applicable

Consent for publication

Not applicable

Availability of supporting data

All the data are included in this article.

Competing interests

The authors declare no competing interests.

Funding

This work was supported by the National Natural Science Foundation of China (Grant No. 62102276), the Natural Science Foundation of Jiangsu Province (Grant No. BK20210705), and the Natural Science Foundation of Educational Commission of Jiangsu Province, China (Grant No. 21KJD520005).

Authors' contributions

Lei Wang, Jianfeng Qu and Tianyu Xu wrote the manuscript; Lei Wang and Jianfeng Qu implemented the model framework and performed the experiment; Wei Chen, Jiajie Xu, Zhixu Li and Lei Zhao provided thoughtful advice to the research.

Acknowledgements

We thank the School of Computer Science and Technology of Soochow University for hardware support.

Authors' information

Lei Wang, Soochow University, lwangcs@stu.suda.edu.cn
Jianfeng Qu, Soochow University, jfqu@suda.edu.cn
Tianyu Xu, Soochow University, tyxutianyu@stu.suda.edu.cn
Zhixu Li, Fudan University, zhixuli@fudan.edu.cn
Wei Chen, Soochow University, robertchen@suda.edu.cn
Jiajie Xu, Soochow University, xujj@suda.edu.cn
Lei Zhao, Soochow University, zhaol@suda.edu.cn

References

[1] Mintz, M., Bills, S., Snow, R., Jurafsky, D.: Distant supervision for relation extraction without labeled data. In: ACL/IJCNLP, pp. 1003–1011 (2009)

[2] Han, X., Gao, T., Lin, Y., Peng, H., Yang, Y., Xiao, C., Liu, Z., Li, P., Zhou, J., Sun, M.: More data, more relations, more context and more openness: A review and outlook for relation extraction. In: AACL/IJCNLP, pp. 745–758 (2020)

[3] Han, X., Zhu, H., Yu, P., Wang, Z., Yao, Y., Liu, Z., Sun, M.: Fewrel: A large-scale supervised few-shot relation classification dataset with state-of-the-art evaluation. In: EMNLP, pp. 4803–4809 (2018)

[4] Finn, C., Abbeel, P., Levine, S.: Model-agnostic meta-learning for fast adaptation of deep networks. In: ICML. Proceedings of Machine Learning Research, vol. 70, pp. 1126–1135 (2017)

[5] Munkhdalai, T., Yu, H.: Meta networks. In: ICML. Proceedings of Machine Learning Research, vol. 70, pp. 2554–2563 (2017)

[6] Dong, B., Yao, Y., Xie, R., Gao, T., Han, X., Liu, Z., Lin, F., Lin, L., Sun, M.: Meta-information guided meta-learning for few-shot relation classification. In: COLING, pp. 1594–1605 (2020)

[7] Snell, J., Swersky, K., Zemel, R.S.: Prototypical networks for few-shot learning. In: NIPS, pp. 4077–4087 (2017)

[8] Ye, Z., Ling, Z.: Multi-level matching and aggregation network for few-shot relation classification. In: ACL (1), pp. 2872–2881 (2019)

[9] Gao, T., Han, X., Liu, Z., Sun, M.: Hybrid attention-based prototypical networks for noisy few-shot relation classification. In: AAAI, pp. 6407–6414 (2019)

[10] Han, J., Cheng, B., Lu, W.: Exploring task difficulty for few-shot relation extraction. In: EMNLP (1), pp. 2605–2616 (2021)

[11] Vinyals, O., Blundell, C., Lillicrap, T., Kavukcuoglu, K., Wierstra, D.: Matching networks for one shot learning. In: NIPS, pp. 3630–3638 (2016)

[12] Devlin, J., Chang, M., Lee, K., Toutanova, K.: BERT: pre-training of deep bidirectional transformers for language understanding. In: NAACL-HLT (1), pp. 4171–4186 (2019)

[13] Soares, L.B., FitzGerald, N., Ling, J., Kwiatkowski, T.: Matching the blanks: Distributional similarity for relation learning. In: ACL (1), pp. 2895–2905 (2019)

[14] Liu, J., Song, L., Qin, Y.: Prototype rectification for few-shot learning. In: ECCV (1). Lecture Notes in Computer Science, vol. 12346, pp. 741–756 (2020)

[15] Paszke, A., Gross, S., Massa, F., Lerer, A., Bradbury, J., Chanan, G., Killeen, T., Lin, Z., Gimelshein, N., Antiga, L., Desmaison, A., Köpf, A., Yang, E.Z., DeVito, Z., Raison, M., Tejani, A., Chilamkurthy, S., Steiner, B., Fang, L., Bai, J., Chintala, S.: Pytorch: An imperative style, high-performance deep learning library. In: NeurIPS, pp. 8024–8035 (2019)

[16] Gao, T., Han, X., Zhu, H., Liu, Z., Li, P., Sun, M., Zhou, J.: Fewrel 2.0: Towards more challenging few-shot relation classification. In: EMNLP/IJCNLP (1), pp. 6249–6254 (2019)

[17] Yang, K., Zheng, N., Dai, X., He, L., Huang, S., Chen, J.: Enhance prototypical network with text descriptions for few-shot relation classification. In: CIKM, pp. 2273–2276 (2020)

[18] Wang, Y., Bao, J., Liu, G., Wu, Y., He, X., Zhou, B., Zhao, T.: Learning to decouple relations: Few-shot relation classification with entity-guided attention and confusion-aware training. In: COLING, pp. 5799–5809 (2020)

[19] Han, J., Cheng, B., Nan, G.: Learning discriminative and unbiased representations for few-shot relation extraction. In: CIKM, pp. 638–648 (2021)

[20] Han, Y., Qiao, L., Zheng, J., Kan, Z., Feng, L., Gao, Y., Tang, Y., Zhai, Q., Li, D., Liao, X.: Multi-view interaction learning for few-shot relation classification. In: CIKM, pp. 649–658 (2021)

[21] Sung, F., Yang, Y., Zhang, L., Xiang, T., Torr, P.H.S., Hospedales, T.M.: Learning to compare: Relation network for few-shot learning. In: CVPR, pp. 1199–1208 (2018)

[22] Fan, M., Bai, Y., Sun, M., Li, P.: Large margin prototypical network for few-shot relation classification with fine-grained features. In: CIKM, pp. 2353–2356 (2019)

[23] Ren, H., Cai, Y., Chen, X., Wang, G., Li, Q.: A two-phase prototypical network model for incremental few-shot relation classification. In: COLING, pp. 1618–1629 (2020)

[24] Geng, X., Chen, X., Zhu, K.Q., Shen, L., Zhao, Y.: MICK: A meta-learning framework for few-shot relation classification with small training data. In: CIKM, pp. 415–424 (2020)

[25] Yang, S., Zhang, Y., Niu, G., Zhao, Q., Pu, S.: Entity concept-enhanced few-shot relation extraction. In: ACL/IJCNLP (2), pp. 987–991 (2021)

[26] Li, X., Sun, Q., Liu, Y., Zhou, Q., Zheng, S., Chua, T., Schiele, B.: Learning to self-train for semi-supervised few-shot classification. In: NeurIPS, pp. 10276–10286 (2019)

[27] Ren, M., Triantafillou, E., Ravi, S., Snell, J., Swersky, K., Tenenbaum, J.B., Larochelle, H., Zemel, R.S.: Meta-learning for semi-supervised few-shot classification. In: ICLR (Poster) (2018)

[28] Liu, Y., Lee, J., Park, M., Kim, S., Yang, E., Hwang, S.J., Yang, Y.: Learning to propagate labels: Transductive propagation network for few-shot learning. In: ICLR (Poster) (2019)