2021-10

Target Unbiased Meta-Learning for Graph Classification

Ming, L

10.1093/jcde/qwab050
Journal of Computational Design and Engineering
Elsevier

Journal of Computational Design and Engineering, 2021, 8(5), 1355–1366

https://doi.org/10.1093/jcde/qwab050
Journal homepage: www.jcde.org

RESEARCH ARTICLE

Target unbiased meta-learning for graph classification

Ming Li[1], Shuo Zhu[2], Chunxu Li[1,2,*] and Wencang Zhao[1,*]

[1]Qingdao University of Science and Technology, Qingdao, Shandong 266042, China and [2]Center for Robotics and Neural Systems, Plymouth University, Drake Circus, Plymouth PL4 8AA, UK

*Corresponding author. E-mail: chunxu.li@plymouth.ac.uk; zhao_coinslab@outlook.com

Abstract

Even though numerous works focus on the few-shot learning issue by combining meta-learning, there are still limits to traditional graph classification problems. The antecedent algorithms directly extract features from the samples, and do not take into account the preference of the trained model to the previously "seen" targets. In order to overcome the aforementioned issues, an effective strategy with training an unbiased meta-learning algorithm was developed in this paper, which sorted out problems of target preference and few-shot under the meta-learning paradigm. First, the interactive attention extraction module as a supplement to feature extraction was employed, which improved the separability of feature vectors, reduced the preference of the model for a certain target, and remarkably improved the generalization ability of the model on the new task. Second, the graph neural network was used to fully mine the relationship between samples to constitute graph structures and complete image classification tasks at a node level, which greatly enhanced the accuracy of classification. A series of experimental studies were conducted to validate the proposed methodology, where the few-shot and semisupervised learning problem has been effectively solved. It also proved that our model has better accuracy than traditional classification methods on real-world datasets.

Keywords: meta-learning; graph neural networks; graph classification; few-shot learning

1 Introduction

Even though the algorithms based on deep learning have powerful feature extraction and knowledge expression abilities (Silver *et al.*, 2016; Li *et al.*, 2017; Devlin *et al.*, 2018; Cai *et al.*, 2021), there are still challenges (Li *et al.*, 2019a; Deng *et al.*, 2021a). On the one hand, deep learning methods fully depend on datasets that require considerable labeled samples for training purposes. Conversely, there are plenty of untagged data that need to be artificially tagged, which is expensive and time consuming (a few-shot learning problem). On the other hand, deep learning is merely for a specific task. In other words, models are only trained for the current tasks. Hence, the performance of trained models needs to be promoted while doing new tasks that have never been seen. In addition, owing to lacking training samples, the trained models may work well on the training set but encounter parameters overfitting problems on the testing set. Therefore, another key point of solving the few-shot learning problem is to overcome the problems of overfitting. Due to these inherent defects of deep learning, it remains gravely difficult to realize artificial intelligence by using deep learning alone.

Meta-learning, or "Learning to learn," has aroused great interest. Unlike traditional artificial intelligence methods, the purpose of meta-learning is to improve the learning algorithm itself, and take into account the experience of multiple learning. This meta-learning paradigm provides an opportunity to solve many traditional challenges of deep learning, including data and computational limitations, as well as the basic problems of generalization. In addition, it is not to learn how to solve a specific task, but to master the learning itself through the tasks that have been learned. Although meta-learning is also a kind of machine learning, there exists a difference, where the goal of the latter is to allow machines to learn, while meta-learning is to allow machines to learn how to learn. Meta-learning, in general, makes extensive use of prior knowledge and experience to guide the

Received: 31 May 2021; **Revised:** 5 July 2021; **Accepted:** 16 July 2021

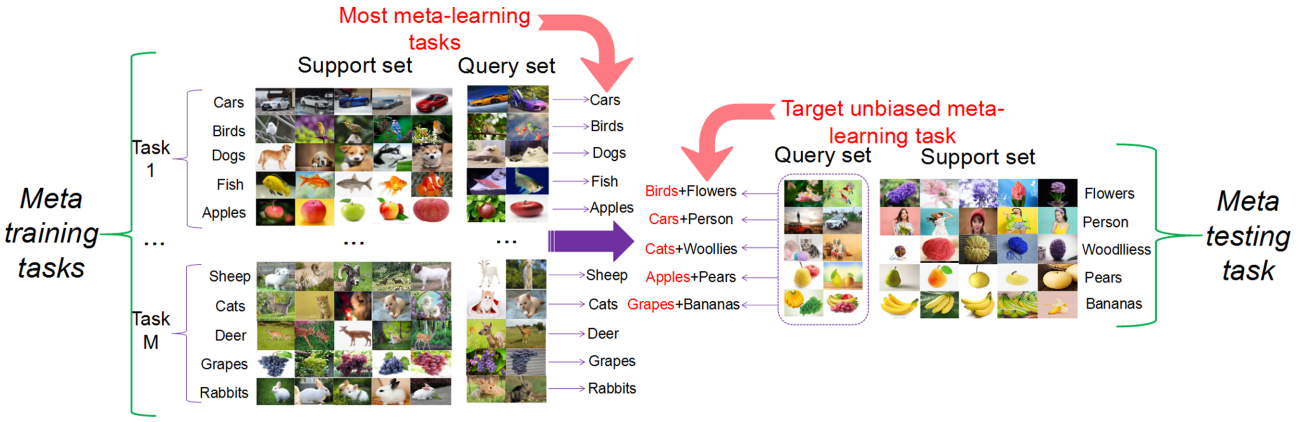

Figure 1: The overview of meta-learning tasks and target bias. To simulate a meta-testing task, we derive M five-way five-shot tasks for meta-training. What is worth noting is that not only do the samples of the query set used for meta-testing contain the classification targets (flower, person, woollies, pears, and bananas), but also include the targets (birds, cars, cats, apples, and grapes) seen in the meta-training stage. Under these circumstances, the trained model is more inclined to identify the target that has been seen and produce the target bias problem that will affect classification accuracy.

learning of new tasks and has the ability to "Learning to learn." What proves beyond dispute is that only if machines have this ability can artificial intelligence be truly realized. Therefore, meta-learning has become a new direction of conquest.

However, although the existing meta-learning methods have achieved excellent results with broad prospects, they all extract feature vectors from the support set and the query set separately, which fail to consider the difference between the sample targets in the support and query set (Pfahringer *et al.*, 2000). The emerging meta-learning method can be used with the training set to derive multiple few-shot classification tasks and simulate classification tasks with a few labeled samples of the testing set. However, when most existing meta-learning methods extract feature vectors, they only take the benefits of making full use of data information caused by prior knowledge into account, but fail to consider the preference of the training model for targets that have already been seen. Also, meta-learning tasks coexist with multiple tasks; they have a certain memory ability for prior knowledge. It will lead to the extracted feature vectors not conducive to the current classification task if this preference is not considered, and we define this preference as "target bias" (representing the preference of the model for the meta-training targets that have been seen in the process of training the classification model). As illustrated in Fig. 1, the labeled dataset in the training task is images of cars, birds, etc. The testing task, unfortunately, keeps classifying flowers, people, etc. If there are cars or birds in the image to be classified besides flowers or persons, the trained model may be more inclined to regard cars or birds as classification targets, other than flowers or persons, which may lead to deviations in the classification results. In order to alleviate this deviation, this paper proposes an interactive attention extraction module in the feature extraction process, which has been exploited to improve the separability of feature vectors and reduce the model's preference for a certain target. It is worth noting that our interactive attention extraction module can be used as a supplement to any feature extraction module, reducing the model's preference of the "seen" targets to improve the classification accuracy.

Also, most of the data in the real-world have complex interactive relationships, which can be represented by graph structures, and graph neural networks (GNNs) are created to fully explore the relationships between samples. In order to make full use of the relationships between samples, a neural network has been introduced as a classification module to map the image-level classification task to the node-level, and obtain the implicit relationship between the samples through the node feature updating module and the edge feature updating module, where each feature vector was treated as a node, and the GNN was utilized to fully express the relationship in between. Simultaneously, in order to solve the issues that the distribution of samples cannot truly represent the distribution characteristics of real data triggered by few training samples, and to further deal with the amount of data that is incapable of meeting the requirements of training and learning, we took the advantages of semisupervised learning (Zhu & Goldberg, 2009) and transductive inference (Joachims *et al.*, 1999; Liu *et al.*, 2018) to enhance the effect.

In summary, our contributions are three-fold:

First, a simple and well-behaved target unbiased meta-learning method for graph classification has been developed; it operates directly on graphs and can quickly lock the target domain of the current classification task and solves the target bias problem of other graph classification methods.

Second, we demonstrate how this form of a graph-based neural network model can be used for few-shot classification of nodes in a graph.

Third, we demonstrate the superior performance of the proposed approach on the standard few-shot learning benchmark datasets and consider new few-shot learning settings (such as transductive and semisupervised learning) that are also important real-world use cases and results show that our approach can outperform baseline algorithms under these settings too.

2 Related Works

Aiming at solving the problem of few-shot learning, a large number of recent studies have focused on meta-learning, because it can quickly adapt to new tasks and transfer useful knowledge between tasks with fewer samples. Models and algorithms of meta-learning, in a general way, can be basically divided into the following three types: optimization-based methods, metric learning-based methods, and memory-based methods. The following is a specific introduction to these distinguished methods.

The optimization-based methods are designed to quickly update the parameters on scarce samples through the design of the model structure, and directly establish the mapping function

between the input and the predicted value. Chelsea Finn and Sergey Levine proposed the MAML algorithm in 2017 (Finn *et al.*, 2017), which is currently one of the most elegant and promising meta-learning algorithms. In 2018, OpenAI released the simple meta-learning algorithm Reptile (Nichol *et al.*, 2018), which repeatedly samples a task, performs stochastic gradient descent (SGD), and updates the initial parameters until the final parameters are learned. The authors of Jamal and Qi (2019) successfully train a task-agnostic meta-learning algorithm, alleviating the problem of task preference. Research (Guo & Cheung, 2020) proposes a few-shot learning algorithm (AWGIM) based on weight generation. The above methods have achieved extreme success in tackling the problem of few-shot learning. Unfortunately, due to the limitations of optimizer selection (such as SGD, Adam, etc.) and learning rate, optimization-based meta-learning methods usually have to update multiple steps to reach a better point, so when models face new tasks, the learning process remains extremely slow. Worse, they are easily overfit when updating the weights in the case of train.

The basic idea of methods based on metric learning is to compare the distance between query set samples and support set samples, and then utilize the means of nearest neighbors to achieve classification. The research works (Sung *et al.*, 2018; Li *et al.*, 2019b; Deng *et al.*, 2021b) are inspired by the Siamese Network (Koch *et al.*, 2015; Guo *et al.*, 2017), the Matching Network (Vinyals *et al.*, 2016), and the Prototypical Network (Snell *et al.*, 2017) and then use metric learning ideas to implement few-shot classification tasks. Metric-based meta-learning methods, also known as non-parametric methods, perform well for few-shot classification, but their effectiveness in other meta-learning fields, such as regression or reinforcement learning, has yet to be proven. Also, when the differences between the testing and training tasks are evident, the effect fails to appear. Furthermore, when the task becomes larger, the pairwise comparison may result in expensive computational costs.

Memory-based methods are to construct an external memory, introduce prior knowledge into it, and then use it to realize the classification of few-shot. The MetaNet (Munkhdalai & Yu, 2017) proposed in ICML2017 is a method that realizes rapid parametrization of generalization tasks. CVPR2018 proposes a few-shot learning algorithm MM-Net (Cai *et al.*, 2018) based on external memory. The innovation of this article is to use external memory modules to achieve few-shot learning tasks. Memory-based methods are one of the most commonly used methods, and have been implemented in few-shot classification, regression tasks, and meta-reinforcement learning. Although these methods are more flexible, due to the learner networks needing to design a learning strategy from scratch, they are slightly less efficient than other meta-learning methods. In addition, they impose restrictions on the model (RNN), which may prevent its development and application to some extent.

The above meta-learning methods have completely affected an immense number of learning tasks, from image classification, video processing to speech recognition, natural language processing, etc. Data, on the other hand, are non-linear and expressed as a graph structure of complex relationships and interdependence in between (Bronstein *et al.*, 2017; Song *et al.*, 2021). Hence, quite a few indispensable operations such as convolution are not suitable for graph data (Deng *et al.*, 2020b; Wu *et al.*, 2020). To sort it out, the GNN is used to embed data into a suitable Euclidean space. Victor Garcia *et al.* used the few-shot learning algorithm of GNN to transfer the distance measurement from Euclidean space to non-European space (Garcia & Bruna, 2017), and calculated the relationship between images by graph model, although the thought is very simple, but obtaining an advanced effect. Moreover, some new network architectures have been proposed recently, such as Graph attention networks (Veličković *et al.*, 2017), Graph generative networks (Zhang *et al.*, 2019), and Graph spatial-temporal networks (Luo & Yuille, 2019). In addition, Luo *et al.* (2020) proposed a deep relationship network to capture the relationship between different samples and constructed a knowledge graph by linking images with tags; Lin *et al.* (2020) proposed a network (HOSP-GNN) for few-shot learning, which can not only use relative metrics in multiple samples to describe higher order structural relationships, but also re-formulate the update rules of the graph structure through alternate calculations between vertices and edges based on higher order structures; Ding *et al.* (2020) introduced a new graph element learning framework-Graph Prototype Network to solve the problem of the few-shot node classification on the attribute network; Ma *et al.* (2020) proposed a direct method that using good initial value to capture the substructure. This paper focuses on the classification of few-shot learning and proposes a new framework AS-MAML. What is worth noting is that Yanbin Liu *et al.* (Sung *et al.*, 2018) proposed transductive method carries on the label propagation to the training set and the testing set in the episode, by establishing an undirected graph together with all the unlabeled data and the annotated data, solved the problem that the classifier is unreliable, due to the lack of training data. Compared with inductive inference methods, it has achieved advanced results. In this paper, similar transductive inference methods are used to make full use of the data without labels. The experimental results show that the classification ability of the model is improved after the addition of transductive inference. However, these methods also have the following shortcoming: They only take the benefits of making full use of data information caused by prior knowledge into account, but fail to consider the preference of the training model for samples that have already been seen. To a certain extent, it will interfere with the model's classification task. In order to alleviate this deviation, this paper proposes an interactive attention extraction module.

Considering the nature of metric learning, the difference between our work and the above methods is that, through employing the interactive attention extraction module during the process of feature extraction, the trained model is able to quickly locate the target to be identified, and naturally reduces the interference of other objects in the same situation. Furthermore, in order to improve the accuracy of classification, the advanced GNN is therefore combined as the classification module. The specific implementation process is described in Section 4.

3 Problem Definition

Meta-learning: It introduces a series of concepts, including *N*-way *K*-shot, meta-training, meta-testing, base class and novel class, support set and query set, etc. Wherein base class and novel class represent the source domain data used in the meta training phase and the target domain data to be learned in the meta testing stage, respectively, in which base class and novel class have no intersection (Santoro *et al.*, 2016; Mishra *et al.*, 2017; Gidaris & Komodakis, 2019; Deng *et al.*, 2020a). The so-called *N*-way *K*-shot classification problem means that the samples in the testing set desired to be classified are all in *N* categories, and only *K* samples in each category are labeled. What to prove is that using these $N \times K$ samples to realize the classification of unlabeled samples in the testing set.

It is known that meta-learning demands learning many tasks to learn a model algorithm with strong generalization ability. Therefore, meta-training is conducted by constructing multiple classification tasks similar to the testing set to ensure the adaptability of the model. Since meta-learning uses tasks as its own training data, the actual training set and testing set of meta-learning are composed of multiple tasks. In order to distinguish, the training set within each task was renamed to support set $S = \{(X_1, Y_1), \ldots, (X_n, Y_n)\}$, where $n = N \times K$, and the testing set was renamed to query set $Q = \{x_1, \ldots, x_q\}$. The formation process of the support set and the query set is: each iteration randomly selects N classes from all the classes in the training set, extracts K samples from all the samples in each class as the support set, and the rest as the query set. The process of meta-training is to use the samples in the support set to train the model. Moreover, the process of meta-testing is to randomly select a few instances from the query set, and let the model query which category it belongs to in the support set, namely the label Y_i, where $i \in (1, n)$; if the classification is correct, it can be proved that the model performance is excellent. Repeating the above meta-learning process can finally get the accuracy of the task model, which will be used to further optimize the parameters of the model. When the training tasks are all completed, the trained model will be applied to classify the samples in the testing set to achieve fine-tuning of the model, which will be used to classify the samples with unknown labels in the testing set (Triantafillou *et al.*, 2019). As illustrated in Fig. 1, they are five-way five-shot tasks, and the number of meta-training tasks is M that is formed by repeating the above meta-learning process M times.

Graph classification: In the real world, there are complex interactive relationships between most data, and this relationship can be represented by a graph structure, which is composed of nodes and edges, where the nodes represent the objects of task processing and edges represent the relationship between them. In this paper, we regard the support set categories or query set samples cascaded with tags as nodes, and the similarity among categories or between categories and query set samples as edges, and the matrix composed of the weights on the edges is regarded as an adjacency matrix (see Section 4 for details). That is to say, our classification task converts the comparison methods at the image level into the comparison method at the graph (node) level. The purpose is to compose the graph structure through the image, and use the GNN to update the node features and edge features to fully mine the relationship between them, thereby increasing the accuracy rate.

Interactive: Most of the existing meta-learning methods are multitasking and independently extracting the feature vectors of the training set and the testing set samples, which leads to the low resolution of the extracted features (the extracted feature vectors are easily affected by prior experience and blindly treat the trained targets as the targets of the current classification task), while our model can extract the interactive attention R^c (which represents the relationship between the prototype and the query set sample feature map) and R^x (which represents the relationship between the query set sample feature map and the prototype) between the training set samples and the testing set samples, making the feature vectors more favorable for classification. In short, the "interaction" in this paper means that the feature extraction network no longer independently extracts the feature vectors of the support set and the query set samples. Instead, the feature vectors related to the current task are generated by extracting the interaction between support set samples and query set samples (locking the target domain quickly through interactive attention).

Bias: Most meta-learning tasks coexist with multiple tasks, and they have a certain memory ability for prior knowledge. As a result, if the trained model's preference for the training target is not taken into account, the extracted feature vectors will be unsuitable for the current classification task, and we refer to this preference as "target bias.". Also, our proposed interactive attention extraction module can quickly lock the classification target by generating interactive attention, so that the meta-learning model reaches target unbiased (we still retain the prior knowledge while interfering with the past goal; this part is shown in Section 4 Methods of the revised manuscript). In short, bias represents the preference of the model for the classification targets (meta-training targets) that have been seen in the process of training the classification model.

4 Methods

From the theory of previous sections, we have known that meta-learning task is composed of multiple N-way K-shot tasks, and this section will expound the execution process of a certain meta-learning task, and the theory of Section 3 has explained that the GNN contains a node update module and an edge update module, and in this section we shall explain that how the distance between the sample and categories be continuously updated to obtain the final similarity through these two modules. We shall first briefly describe how the input context is mapped into a target unbiased feature vector, and then particularize the related architecture. Also, our proposed model is divided into three modules to be described: feature extraction module, interactive attention extraction module, and graph classification module. The visualization of the operation process is shown in Algorithm 1 (Figs. 2 and 3).

Feature extraction module: We use the traditional prototype network (Snell *et al.*, 2017) to extract the original feature vectors. In the same way, the entire support set $S = \{(X_1, Y_1), \ldots, (X_n, Y_n)\}$, where $n = N \times K$, and query set samples $Q = \{x_1, \ldots, x_q\}$ are used as the inputs of the embedded function f_ϕ. The class feature mapping (prototype) C_i (equation 1), where $i = 1, \ldots, N$, of the support set S and feature mapping $f_\phi(x_j)$ of query set samples x_j, where $j \in (1, q)$, are outputs:

$$C_i = \frac{1}{|K|} \sum_{k=1}^{K} f_\phi(X_i^k) \tag{1}$$

where $i = 1, \ldots, N$, C_i represents the class center of the i-th class in support set, K represents the number of labeled samples in support set, and $f_\phi(X_i^k)$ indicates the feature mapping of the k-th sample of the i-th class in support set.

The network structure that realizes the above functions consists of four convolutional blocks, and each convolutional block includes a convolutional layer, a batch normalization layer, and a ReLU activation function layer. Its intention is to utilize a simple four-level network structure to map the input information to a high-dimensional feature space, i.e. $R^D \to R^M$, where $M = c \times h \times w$, with c, h, and w representing the number of channels and the height and width of the feature map, respectively. Also, the mean value of its high-dimensional feature vectors is taken as prototype C_i. The ReLU activation function is made use of nonlinear mapping. The visualization is shown in Fig. 4.

Interactive attention extraction module: In order to further process the original feature mappings and obtain the unbiased target feature vectors, the interactive attention extraction module is introduced. First, the prototypes $C_i \in R^{c \times h \times w}$ of support set and the feature mappings $f_\phi(x_j) \in R^{c \times h \times w}$ of query set samples

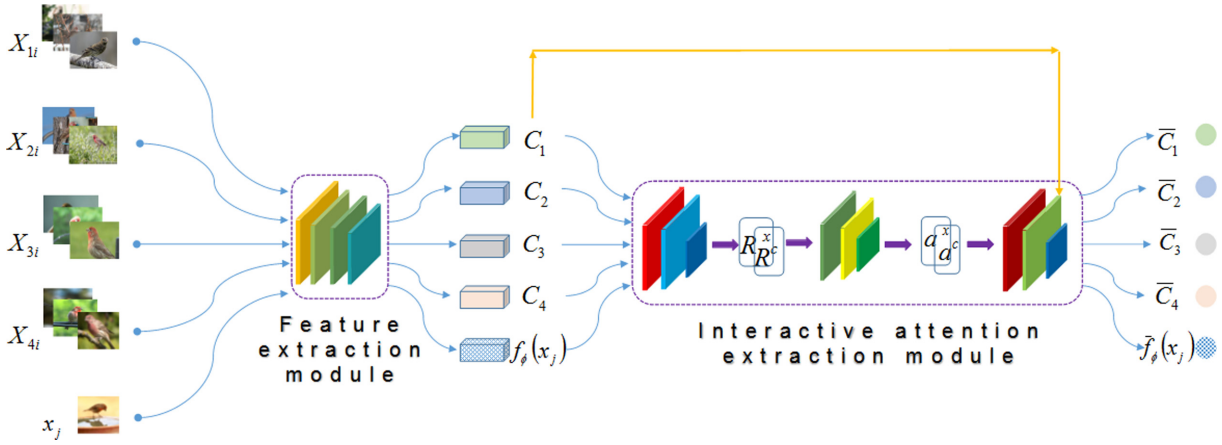

Figure 2: Overview of unbiased feature vectors extraction process in a meta-learning task. The feature extraction module has been used to acquire class center feature vectors (C_1, C_2, C_3, and C_4) of support set and sample feature vectors $[f_\phi(x_j)]$ $(j = 1, \ldots, q)$ of query set. Afterwards, interactive attention extraction module is used to obtain unbiased feature mappings $\bar{C}_i$ and $\bar{f}_\phi(x_j)$.

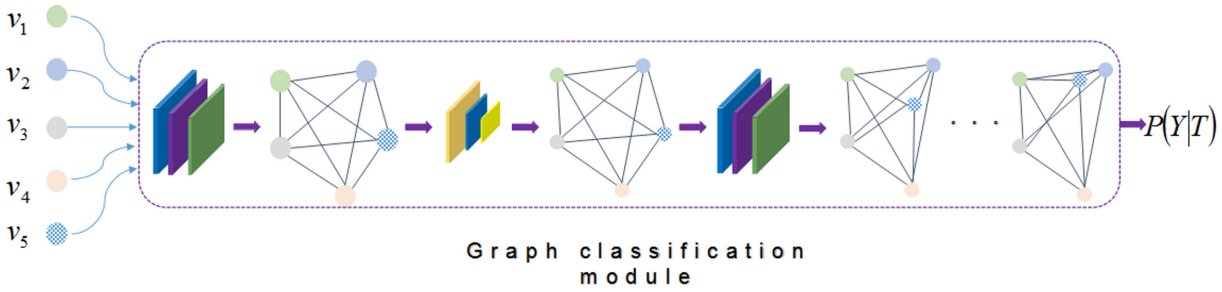

Figure 3: Overview of unbiased feature vectors classification process in a meta-learning task. The graph classification module has been used to update feature vectors of nodes and adjacency matrix, ultimately, obtains the similarity scores between nodes.

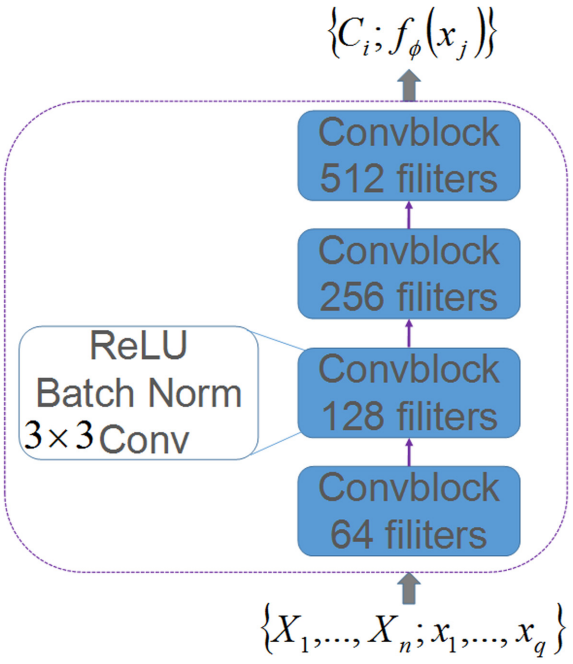

Figure 4: Overview of the network structure of feature extraction module.

are mapped to $R^{c \times m}$, where $m = h \times w$, using a multilayer neural network. At this time, m represents the numbers of spatial positions on each feature mapping. The feature mappings extracted by the prototype network now become: $C_i \to (c_1, \ldots, c_m) \in R^{c \times m}$, $f_\phi(x_j) \to (x'_1, \ldots, x'_m) \in R^{c \times m}$ (where $c_p|_{p=1}^m$ and $x'_q|_{q=1}^m$ represent the class feature vectors and the query sample feature vectors of the p-th and q-th spatial position, respectively), and then we use equations (2) and (3) to calculate the interaction relationship mapping $R^c \in R^{m \times m}$ and $R^x \in R^{m \times m}$:

$$R_p^c = \left(\frac{c_p}{\|c_p\|_2}\right)^T \left(\frac{x'_q}{\|x'_q\|_2}\right) \in (r_1^c, \ldots, r_m^c), \tag{2}$$

where $p \in (1, \ldots, m)$, $q = 1, \ldots, m$, and c_p and x'_q represent the class feature vectors and the query sample feature vectors of the p-th and q-th spatial position, respectively.

$$R_q^x = \left(\frac{x'_q}{\|x'_q\|_2}\right)^T \left(\frac{c_p}{\|c_p\|_2}\right) \in (r_1^x, \ldots, r_m^x), \tag{3}$$

where $p = 1, \ldots, m$, $q \in (1, \ldots, m)$, and c_p and x'_q represent the class feature vectors and the query sample feature vectors of the p-th and q-th spatial position, respectively.

Now the interactive attention mappings can be obtained, where $R^c = (r_1^c, \ldots, r_m^c) \in R^{m \times m}$ represent the relationship between the prototype C_i and the query set sample feature mapping $f_\phi(x_j)$, and the attention mappings $R^x = (r_1^x, \ldots, r_m^x) \in R^{m \times m}$ represent the relationship between the query set sample feature mapping $f_\phi(x_j)$ and the prototype C_i. Furthermore, from the

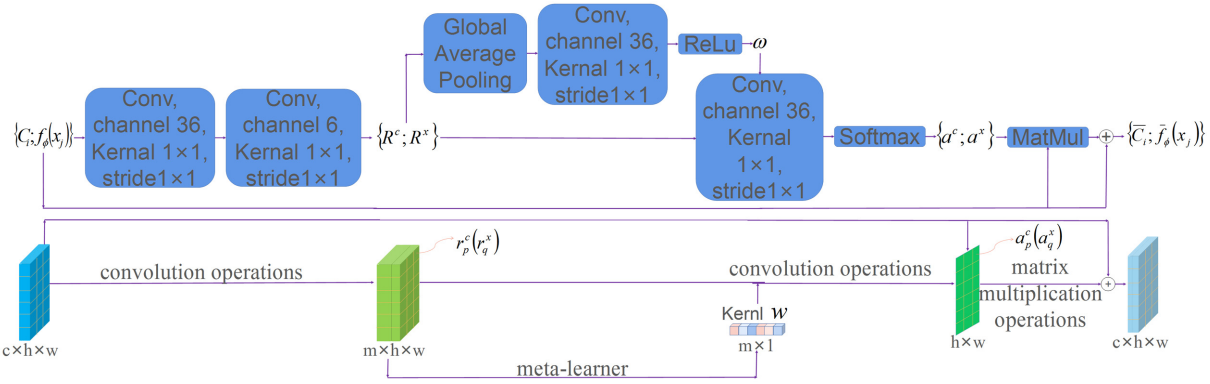

Figure 5: Overview of the network structure of the interactive attention extraction module.

spatial position, $r_p^c \in R^m$ represents the interaction between the p-th spatial position of class feature mapping c_p and all spatial positions of query sample feature mappings $x'_q|_{q=1}^m$. Similarly, $r_q^x \in R^m$ represents the interaction between the q-th spatial position of query sample feature mapping x'_q and all spatial positions of class feature mappings $c_p|_{p=1}^m$. At this point, the local interaction between class prototypes and query set sample feature vectors are defined by $R^c \in R^{m \times m}$ and $R^x \in R^{m \times m}$, respectively.

Next, convolution blocks are used to convolve $R^c \in R^{m \times m}$, $R^x \in R^{m \times m}$ (convolution kernel function $\omega \in R^m$ with $m \times 1$ size can be generated adaptively by a meta-learner, which will be elaborated in Fig. 5) to convert from vectors to scalars, and a Softmax function limits them keep numbers a_p^c (the interactive attention coefficient, which represents the interaction between the p-th spatial position of class feature mapping c_p and all spatial positions of query sample feature mappings $x'_q|_{q=1}^m$), a_q^x (the interactive attention coefficient, which represents the interaction between the q-th spatial positions of query sample feature mapping x'_q and all spatial position of class feature mappings $c_p|_{p=1}^m$) belonging to (0, 1) (shown as equations 4 and 5). The feature vector of the i-th [where $i \in (1, N)$] support set class with interactive attention coefficient in the p-th [where $p \in (1, m)$] spatial position is recorded as $a_p^c C_i$, and the sample feature vector of the j-th [where $j \in (1, q)$] query set sample in the q-th spatial position [where $q \in (1, m)$] with interactive attention coefficient is recorded as $a_q^x f_\phi(x_j)$.

$$a_p^c \propto \mathbf{exp}\left(\frac{\omega^T r_p^c}{t}\right), \tag{4}$$

where $p \in (1, \ldots, m)$, a^c represents the interactive attention coefficient, and t represents a temperature hyperparameter.

$$a_q^x \propto \exp\left(\frac{\omega^T r_q^x}{t}\right), \tag{5}$$

where $q \in (1, \ldots, m)$, a^x represents the interactive attention coefficient, and t represents a temperature hyperparameter. The temperature hyperparameters in this paper are used to make outputs of the Softmax function smoother, and if the values of these hyperparameters are too large, after Softmax function output, it will lead to greater classification error rate and lower classification accuracy, so the values of temperature hyperparameters should be smaller numbers (less than 1), and we set them to 0.025 in this paper.

We add the vectors C_i and $f_\phi(x_j)$ that contain historical information and the vectors $a_p^c C_i$ and $a_q^x f_\phi(x_j)$ that contain interactive attention, respectively, to generate the target unbiased support set class feature vectors $(1 + a_p^c)C_i$ and the target unbiased query set sample feature vectors $(1 + a_q^x) f_\phi(x_j)$. In brief, the former of $(1 + a_p^c)C_i$ and $(1 + a_q^x) f_\phi(x_j)$ represents the feature vectors containing prior knowledge, and the latter of $(1 + a_p^c)C_i$ and $(1 + a_q^x) f_\phi(x_j)$ represents the feature vectors related to the current classification target. We denote class prototype C_i weighted by $(1 + a_p^c)$ and query set sample feature map $f_\phi(x_j)$ weighted by $(1 + a_q^x)$ as $\bar{C}_i$ and $\bar{f}_\phi(x_j)$, respectively. Up till now, target unbiased feature mappings $\bar{C}_i$ and $\bar{f}_\phi(x_j)$ are acquired, which are more suitable for classifying unseen targets.

Two convolutional blocks have been used to obtain the interactive attention mappings R^c and R^x (the internal structure of the convolution block is shown in Fig. 5), and in order to acquire the interactive attention coefficients a_p^c and a_q^x, first a convolutional block has been utilized to map R^c and R^x to a low-dimensional feature space, and then we use a softmax function to limit them among (0, 1). For the sake of generating convolution kernel adaptively according to the interactive attention mappings R^c and R^x, we first employ a global average pool and map them to the low-dimensional space through the convolution block, and then output the convolution kernel $\omega \in R^m$ by the activation function ReLU. Furthermore, so as to gain the target unbiased feature mappings $\bar{C}_i$ and $\bar{f}_\phi(x_j)$, matrix multiplication operations are used to multiply matrix C_i with matrix a_p^c and matrix $f_\phi(x_j)$ with matrix a_q^x. Then, we use summation operations to let them combine with original feature vectors C_i and $f_\phi(x_j)$, respectively. The network structure of the interactive attention extraction module is shown in Fig. 5.

Graph classification module: In order to improve the accuracy of a classification and finally complete the classification of unknown label samples in the query set at the graph level, G (Garcia & Bruna, 2017) are used to fully express the relationship between support set categories and the query set samples. The GNN is a graph model composed of nodes and edges. We make the target unbiased category prototype $\bar{C}_i$ and the target unbiased query set sample feature vector $\bar{f}_\phi(x_j)$ in the query set concatenate with the one-hot encoding of the label, and regard it as the node of the graph structure and express it as $V = (v_1, \ldots, v_z)$, where $z = N + q$, where N is the number of classes of support set and q is the number of samples in the query set. The weight of each edge represents the similarity between the query set sample and the query set category. The process of obtaining adjacency matrix A from target unbiased feature vector v_i and v_j, where $i, j = 1, \ldots, z$, is shown in equation (6), where each element $A_{i,j}$ of the adjacency matrix is processed by the absolute difference between the class center feature vector and the query set sample feature vector.

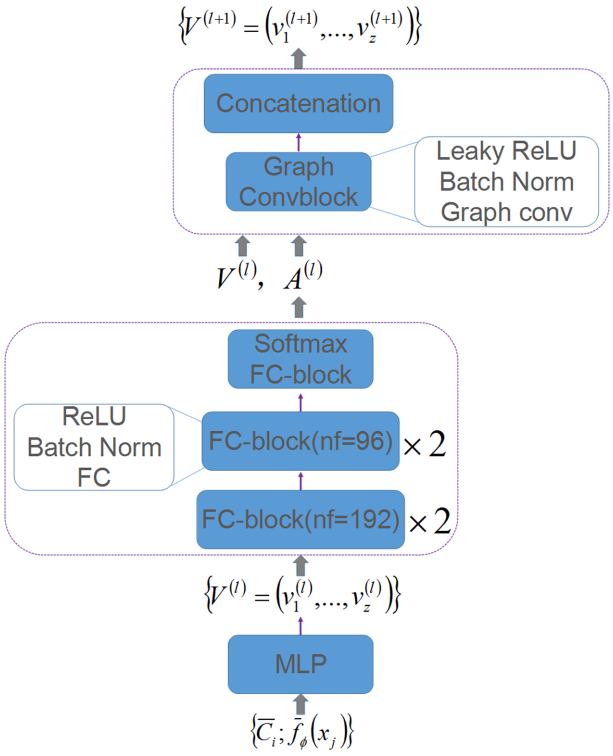

Figure 6: Overview of the network structure of the graph classification module.

The graph classification module is composed of multiple graph nerves. A GNN model is made up of an adjacency matrix (weight) updating module that changes the graph model structure and a node feature updating (graph convolution) module that does not change the graph model structure. Where the weight updating module consists of five fully connected layers, their function is to make similar nodes closer. The output of the GNN model represents the similarity between the query set sample and the support set category.

Graph convolution module consists of graph convolution layer, batch normalization layer, Leaky ReLU activation function layer, and a cascade operation layer. Its function is to utilize the adjacency matrix and node feature vectors of the last layer network to update the node feature vectors of this layer so that it contains global information, which is beneficial to the classification operation. Its visualization diagram is shown in Fig. 6.

$$A_{i,j}^{(l)} = \mathrm{MLP}\left(\mathrm{abs}(v_i^{(l)} - v_j^{(l)})\right), \tag{6}$$

where l represents the l-th layer network, v represents the target unbiased feature vectors, and A represents the adjacency matrix.

Training method: When performing few-shot classification, the model is optimized by minimizing the cross-entropy loss (equation 7). Under semisupervised learning and transductive inference setting, the training process is similar to the few-shot learning process except that the sample label is represented by its K-simplex uniform distribution.

$$L(\bar{f}_\phi(x_j), Y) = -\sum_k Y_k \log P(y_j = Y_k | T), \tag{7}$$

where y_j is the prediction label of the query set sample x_j and $\bar{f}_\phi(x_j)$ represents the target unbiased query set sample feature vector.

5 Experiments

To demonstrate the efficacy and progress of our meta-learning method, three datasets were chosen: Omniglot, MiniImageNet, and TieredImageNet. Multiple sets of experiments have been implemented, including supervised and semisupervised few-shot learning classification, as well as the classification under transductive inference settings. The classification results indicate that our method has an even better performance than state-of-the-art methods.

5.1 Datasets

Omniglot contains 1623 different handwritten characters from 50 different letters. Each character is drawn online by 20 different people. In other words, there are 1623 categories, but only 20 images per category in Omniglot. The dataset has been split into 1200 classes for training and the remaining 423 for testing (Lake *et al.*, 2015).

MiniImageNet is taken from ImageNet, including 100 categories; each category contains 600 sample data. Among them, 64 category data are used as the training set, 16 category data as verification set, and 20 category data as testing set (Snell *et al.*, 2017).

TieredImageNet with MiniImagenet is a subset of ILSVRC-12, which represents a larger subset of ILSVRC-12 with 608 classes. As Omniglot groups characters into letters, TieredImageNet classifies categories into broader classes corresponding to higher level nodes in the ImageNet hierarchy. There are a total of 34 large classes. Each class contains 10 to 30 categories. These are divided into 20 training sessions, 6 validation sessions, and 8 testing sessions (Ren *et al.*, 2018).

5.2 Baselines

Matching Net (Vinyals *et al.*, 2016): It is based on cosine distance and proposed by Vinyals O *et al.*, which is innovatively implemented in the form of matching, where a few-shot classification task maps the feature information of the sample to a higher dimensional and more abstract feature space.

Proto Net (Snell *et al.*, 2017): It is based on squared Euclidean distance and proposed by Snell J. *et al.*, which uses squared Euclidean distance as the distance measurement method, instead of the commonly used cosine distance. The usage of the episode minibatch gradient descent training method makes the samples with more categories available during training.

Relation Net (Sung *et al.*, 2018): It is based on the neural network and proposed by Sung F. *et al.*, which learned a learnable non-linear similarity measurement method for few-shot or even one-shot learning tasks.

GNN (Garcia & Bruna, 2017) *and EGNN* (Kim *et al.*, 2019): In order to prove the advanced nature of our feature extraction module, we compared the GNN based on the non-Euclidean domain proposed by Garcia V *et al.* and the EGNN proposed by Jongmin Kim *et al.*; they provided an algorithm that uses GNN to solve few-shot learning, and utilize graph model to calculate the relationship between images.

TPN (Liu *et al.*, 2018): It proposes a few-shot learning algorithm with transductive propagation network (TPN), and the motivation is that our common supervised learning methods consider separately the training set with a sample label and the testing set without a sample label, which caused poor generalization. Transductive learning is to input the labeled training set

Table 1: Few-shot classification results of five-way(%).

Datasets Methods	Taxonomy	Trans.	MiniImageNet		TieredImageNet	
			One-shot	Five-shot	One-shot	Five-shot
Matching Net	Image-level	N	43.56 ± 0.26	55.31 ± 0.73	54.02 ± 0.00	70.11 ± 0.00
Matching Net	Image-level	Y	44.20 ± 0.10	57.00 ± 0.12	56.85 ± 1.84	69.57 ± 0.59
Proto Net	Image-level	N	46.61 ± 0.78	65.77 ± 0.70	48.58 ± 0.00	69.57 ± 0.00
Proto Net	Image-level	Y	49.42 ± 0.78	68.20 ± 0.66	52.64 ± 0.96	73.30 ± 0.71
Relation Net	Image-level	N	50.40 ± 0.80	65.30 ± 0.70	54.48 ± 0.00	71.31 ± 0.00
Relation Net	Image-level	Y	52.40 ± 0.00	65.36 ± 0.00	54.19 ± 0.28	75.31 ± 0.88
GNN	Node-level	N	50.33 ± 0.36	66.41 ± 0.63	63.56 ± 0.84	75.31 ± 0.73
GNN	Node-level	Y	52.78 ± 0.00	66.42 ± 0.00	65.32 ± 0.70	76.58 ± 0.86
EGNN	Node-level	N	52.46 ± 0.45	66.85 ± 0.40	57.94 ± 0.42	70.98 ± 0.40
EGNN	Node-level	Y	63.54 ± 0.75	76.37 ± 0.00	66.41 ± 0.24	80.15 ± 0.00
TPN	Node-level	Y	53.75 ± 0.86	69.43 ± 0.67	57.53 ± 0.96	72.58 ± 0.74
Our method	Node-level	N	71.94 ± 0.94	73.16 ± 0.61	78.09 ± 0.51	82.93 ± 0.63
Our method	Node-level	Y	**73.72 ± 0.95**	**76.37 ± 0.62**	**78.63 ± 0.63**	**84.23 ± 0.66**

and the unlabeled testing set into the network for training, and then predict the results of this part of the testing set.

5.3 Parameter settings

Data preprocessing: In order to test the feasibility of the implementation on three datasets, we set the number of novel categories to five ($N = 5$), and the number of training examples per novel category to one ($K = 1$) or five ($K = 5$), implying that each class has only one or five labeled samples in the support set of meta-training and meta-testing tasks. At the same time, we set the number of testing examples for per novel category when training as six, and the number of testing examples for per novel category when testing as fifteen. Our initial feature extractor denotes four layer blocks of depth 3 with 3×3 kernels. The number of filters in each block is 64, 128, 256, and 512, respectively, and the image size entered is 84×84, and the experimental results are obtained by the settings with 95% confidence intervals. As for transductive setting, we have a chance to access the query data in the inference stage, so the unlabeled set and the query dataset are the same. All ablation experiments are performed under the transduction setup and the network depth of the classification module is 3.

Implementation details: To verify the accuracy of the implementation on three datasets, five-way one-shot and five-way five-shot experiments will be conducted. We fix the values of episodes per epoch when training as 5000, 1200, and 13 980, for Omniglot, MiniImageNet, and TieredImageNet, respectively. Also, we set the number of episodes per epoch when testing as 2000 for three datasets. An SGD is used to optimize the model, where the initial learning rate is set as 0.1 and weight decay is set as 5×10^{-4}; meanwhile, the momentum of the optimization algorithm is set as 0.9; batch size for training and testing is set as 4 and 8, respectively. Also, all experiments in this paper are accelerated training on two NVIDIA GTX 1080Ti GPUs, using Pytorch programming language to train and test in Ubuntu 16.04 environment and the other settings of models are the same as the implementation of their original papers.

5.4 Experimental results

Few-shot learning: With respect to MiniImageNet and TieredImageNet datasets, we conduct five-way classification tasks. In addition, we set the number of labeled samples per class between 1 and 5 ($K = 1$ and 5). Table 1 indicates that our target unbiased meta-learning method immensely outperforms the contrastive methods. Furthermore, the preponderance of transductive inference is capable of being predicted by labeled samples, and able to find clusters by using the information of unlabeled test samples, further classifying them more effectively. To achieve the best generalization ability on these data, we add transductive settings (Y) and compare its results with inductive settings (N). The experimental results show that assuming unlabeled data being equivalent to the data to be tested (transductive) has an advantage over simply training labeled samples (inductive), with accuracy improving by at least 0.54% (78.09% vs 78.63%) in the five-way one-shot setting and 1.30% (82.93% vs 84.23%) in the five-way five-shot setting. Our method belongs to the category of metric learning. In order to prove the advancement of the classification module, we first compare the matching network based on cosine distance proposed by Vinyals O *et al.* Also, the accuracy of our method outperforms it under both five-way one-shot settings and five-way five-shot settings. Second, we compare the prototype network based on squared Euclidean distance proposed by Snell J. *et al.* From Table 1, we can observe that our method outperforms it by at least 4.96% (68.20% vs 73.16%) under the five-way one-shot and five-way five-shot settings. Then, we compare our method to the neural network-based relational network proposed by Sung F. *et al.*, and find that our work improves accuracy significantly. Furthermore, in order to prove the advanced nature of our feature extraction module, we compare our method with the GNN based on the non-Euclidean domain proposed by Garcia V *et al.* and the EGNN proposed by Jongmin Kim *et al.*, and the accuracy of our method improved by at least 7.40% (63.54% vs 71.94%) under the five-way one-shot settings, and it is no worse than EGNN, in addition to improving at most 4.08% (80.15% vs 84.23%) under the five-way five-shot settings with the help of the interactive attention extraction module. Moreover, in the case of the same setting of transductive inference, since our method introduces the interactive attention extraction module, the accuracy rate is improved by at least 6.94% (69.43% vs 76.37%) under five-way one-shot and five-way five-shot settings.

Semisupervised learning: It is introduced to utilize a mass of label-free samples, to obtain valuable information from the data, and to deal with the problem of poor generalization ability of supervised learning as well as inaccurate of unsupervised learning. We perform five-way five-shot semisupervised learn-

Table 2: Semisupervised results of five-way five-shot setting(%).

Datasets Methods	Class	MiniImageNet			
		20%	40%	60%	100%
GNN	S	50.33 ± 0.36	56.91 ± 0.42	61.02 ± 0.28	66.41 ± 0.63
GNN	SS	52.45 ± 0.88	58.76 ± 0.86	62.92 ± 0.57	66.41 ± 0.63
EGNN	S	58.65 ± 0.55	56.91 ± 0.00	62.39 ± 0.00	66.85 ± 0.40
EGNN	SS	62.62 ± 0.00	63.32 ± 0.00	68.58 ± 0.00	76.37 ± 0.00
TPN	SS	59.87 ± 0.00	62.42 ± 0.00	67.36 ± 1.12	68.29 ± 0.62
Our method	S	59.18 ± 0.21	62.98 ± 1.01	**69.85 ± 0.46**	74.37 ± 0.25
Our method	SS	**63.62 ± 0.71**	**64.32 ± 0.24**	68.47 ± 0.59	**76.54 ± 0.89**
		TieredImageNet			
GNN	S	51.36 ± 0.25	55.52 ± 0.85	66.29 ± 0.26	75.31 ± 0.99
GNN	SS	54.45 ± 0.81	56.68 ± 0.26	64.18 ± 0.36	76.85 ± 0.63
EGNN	S	60.12 ± 0.42	64.87 ± 0.14	65.36 ± 0.57	70.98 ± 0.40
EGNN	SS	63.18 ± 0.36	66.24 ± 0.43	70.80 ± 0.94	80.15 ± 0.00
TPN	SS	61.52 ± 0.87	64.32 ± 0.53	69.52 ± 0.57	72.58 ± 0.74
Our method	S	66.41 ± 0.14	71.12 ± 0.72	72.34 ± 0.64	82.93 ± 0.63
Our method	SS	**69.80 ± 0.31**	**71.93 ± 0.56**	**74.56 ± 0.33**	**84.23 ± 0.66**

Algorithm 1: Training strategy of target unbiased meta-learning

Inputs: $S = \{(X_1, Y_1), \ldots, (X_n, Y_n)\}$, $Q = \{x_1, \ldots, x_q\}$, where $n = N \times K$;
Outputs: labels y of images in query set;
For all M tasks **do**:
| Compute C_i, $f_\phi(x_j)$ using equation (1), where $i \in (1, n)$, $j \in (1, q)$;
| Compute R_p^c, R_q^x using equations (2) and (3), where $p \in (1, m)$, $q \in (1, m)$;
| Compute a_p^c, a_q^x using equations (4) and (5), where $p \in (1, m)$, $q \in (1, m)$;
| Compute $\tilde{C}_i$, $\tilde{f}_\phi(x_j)$ using a_p^c, a_q^x;
| Connect $\tilde{C}_i$, $\tilde{f}_\phi(x_j)$ with its labels $\rightarrow V = \{v_1, \ldots, v_z\}$, where $z = N + q$;
| **For** all layers $=1, \ldots, l$ **do**:
| | Compute $A^{(l)}$ using equation (6);
| | Compute $V^{(l+1)}$ using $A^{(l)}$, $V^{(l)}$;
| **End**
| Predict the label y_j of x_j based on A;
| Compute the cross-entropy loss function L using equation (7);
| Output the label y_j, where $j \in (1, q)$, of x_j based on the minimum value of L;
End

ing experiments on two datasets, namely MiniImageNet and TieredImageNet. As shown in Table 2, diverse results are acquired, merely with labeled samples (supervised, which is defined as S) and with labeled as well as unlabeled samples (semisupervised, which is defined as SS), by setting the proportion of labeled data to 20%, 40%, 60%, and 100%. The proposed target unbiased meta-learning method is compared with node-labeling GNN, EGNN, and TPN. Results indicate that our method can also achieve good performance with fewer labeled samples, where the accuracy of our method outperforms GNN by at least 6.73% (52.45% vs 59.18%), 4.22% (58.76% vs 62.98%), and 5.93% (62.92% vs 69.85%) under the 20%, 40%, and 60% settings, respectively. Our method, compared with EGNN, can obtain a margin [1.00% (62.62% vs 63.62%), and 6.62% (63.18% vs 69.80%), when 20% labeled] on MiniImageNet and TieredImageNet, respectively, especially when the labeled portion was small. Simultaneously, when compared with TPN, the accuracy of our method can improve at least 3.75% (59.87% vs 63.62%), 1.90% (62.42% vs 64.32%), and 1.05% (67.36% vs 68.41%) under the 20%, 40%, and 60% settings, respectively. In addition, with the increase of labeled samples, the classification accuracy of various methods remains gradually improved and our method is capable of extracting more useful information from samples compared to baselines, on both inductive and transductive settings.

5.5 Ablation experiments

Impact of the interactive attention extraction module: To demonstrate the effectiveness of the interactive attention extraction module, we visualize the feature mappings of our target unbiased meta-learning method (TUML), squared Euclidean distance-based prototype network (Proto), graph models-based GNN, and TPN using transductive inference instead of inductive inference when training graph modules, and the results are shown in Fig. 7. What is obvious is that when the targets in the sample are chaotic, e.g. the current classification target woollies are mixed with the target cat already seen during training, the baseline methods cannot accurately identify the target domain, and our method can roughly locate the location of woollies. Also, when the target distribution in the sample is relatively clear, the traditional meta-learning methods can identify the target domain, but they are susceptible to interference from the targets that have been seen, and cannot accurately locate the current classification target. For example, if there are existing current classification target bananas as well as target grapes that have been seen in the training process, the feature extraction models of baselines are easily interfered with by grapes, and our method can accurately locate the target domain by introducing the feature extraction module, which is capable of eliminating the graph interference and achieving target unbiased.

Impact of the number of samples per class (K): We quantify how influential it is to add the number of labeled samples per class on the Omniglot. We make K a number between (0, 10), and fix N to be 5. As expected, this change often be beneficial, both for our method and for baselines. From Fig. 8, it can be seen that the difficulty degrades as the shot increases, and more examples per class indeed make it easier to correctly classify that class.

Impact of the number of classes per task (N): To examine the effect of N on test accuracy, we alter the value of N on the Omniglot by taking the integer between (0, 20), and fix K to be 5. What proves evident is that our method is always superior to

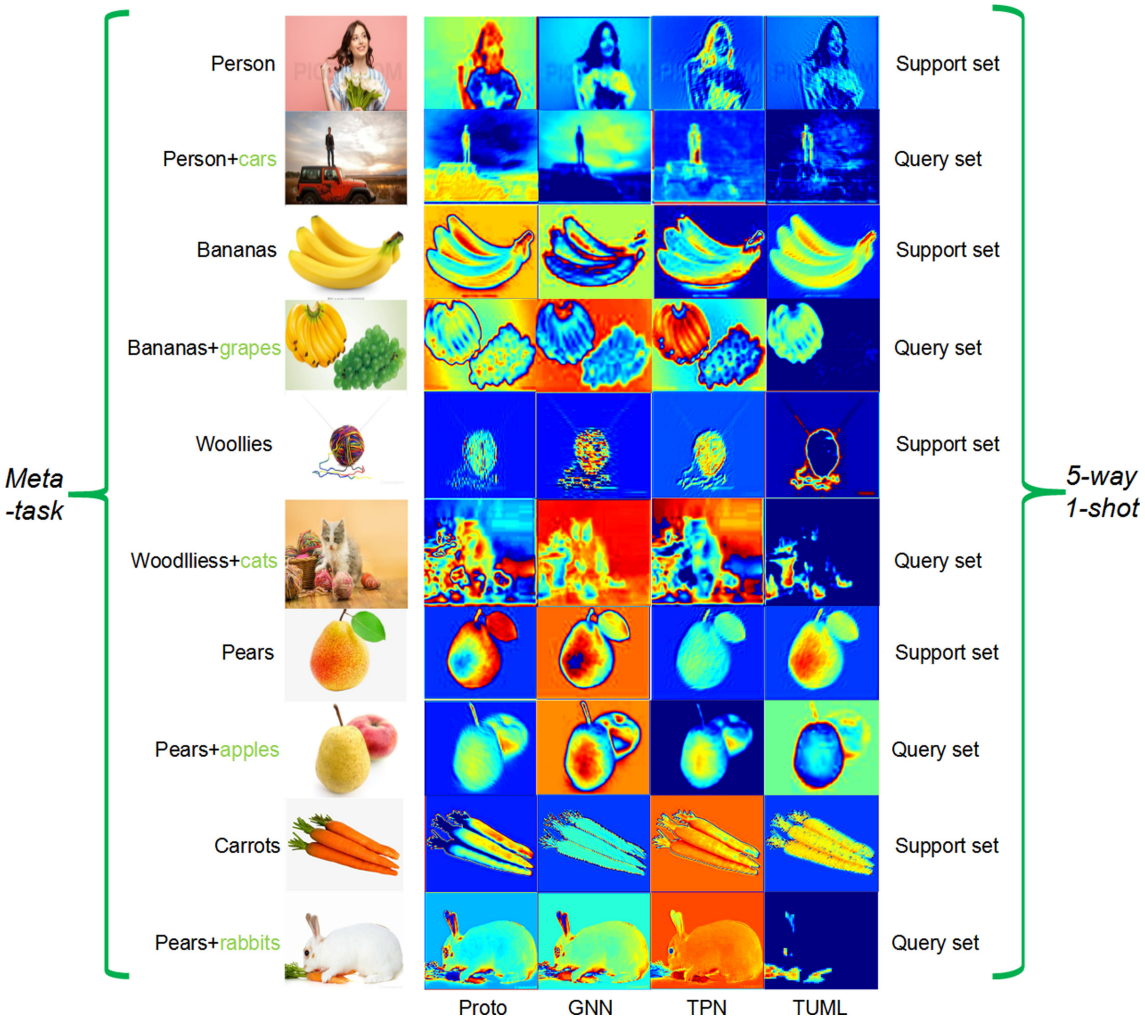

Figure 7: The feature mapping visualization of different methods.

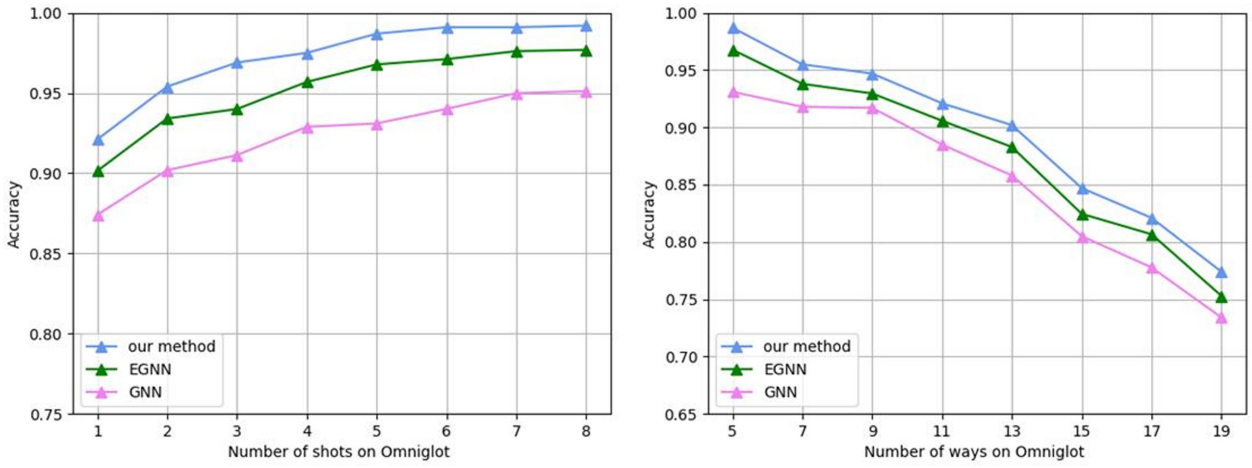

Figure 8: Plots of testing accuracy for different numbers of shots and ways.

baseline methods. As illustrated in Fig. 8, with the increase of N, the prediction accuracy of various methods has decreased, which is due to the incremental number of categories, increases the difficulty of classification, and thus the accuracy of classification decreases.

Impact of the network structure: For further demonstration of the superiority of our approach, we performed ablation experiments of the network structure on the dataset MiniImageNet and TieredImageNet, results as shown in Fig. 9, where P, I, and G represent the prototype network, the interactive attention

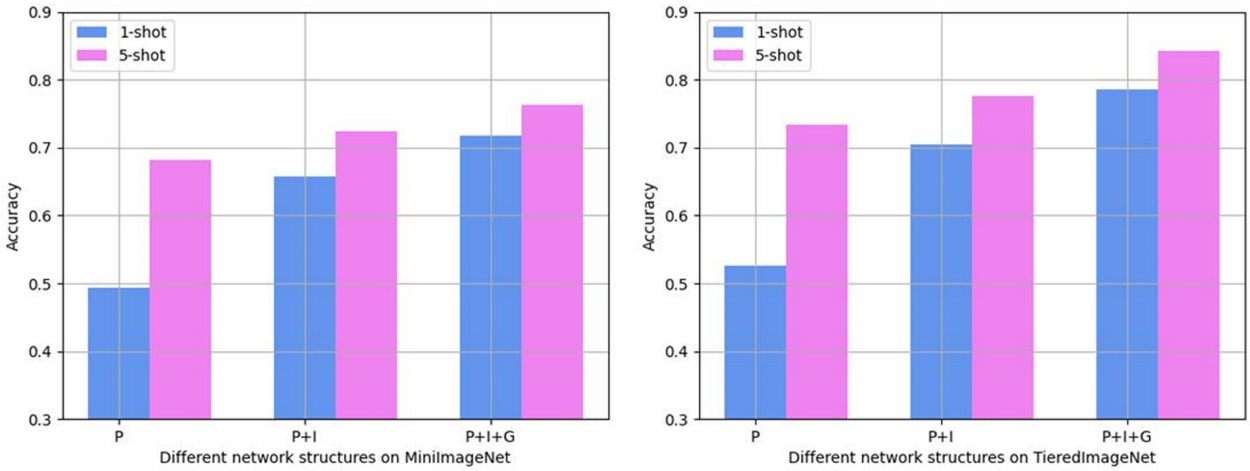

Figure 9: Histogram of testing accuracy for different network structures.

Table 3: Comparisons of the number of parameters and time complexity.

Methods	Trans.	Parameters	Time (sec.)	
			One-shot	Five-shot
Proto Net	N	8232.96K	0.96	1.25
Proto Net	Y	8232.96K	0.97	1.26
Our method	N	8232.96K	1.72	2.02
Our method	Y	8232.96K	1.93	2.25

extraction module, the graph classification module, respectively. First of all, we use ResNet12 to extract features, and then classify them by square Euclidean distance. The accuracy of prototype network is compared with the method adding interactive attention extraction module, and the results certify that the addition of interactive attention extraction module improves the performance of the model by at least 4.22% (68.20% vs 72.42%) on MiniImageNet and 4.30% (73.30% vs 77.60%) on TieredImageNet. Last but not the least, employing GNN as a classifier, the accuracy is able to be improved from 65.78% and 72.42% to 71.72% and 76.37% on MiniImageNet and from 70.51% and 77.60% to 78.63% and 84.23% on TieredImageNet, under one-shot and five-shot settings, respectively.

Complexity analysis: We measure the number of parameters and time complexities for several comparisons on MiniImageNet, and the results are shown in Table 3. We evaluate the average time for a task with 15 query samples per class under five-way one-shot and five-way five-shot settings on the environment of the NVIDIA GTX 1080Ti GPU. Table 3 indicates that there are almost no additional parameters and the time consumption of our method is quite comparable. Specifically speaking, the parameters of the target unbiased feature extraction module are mainly on meta-learner, which aims to train kernel function w and the average time consumption of this module is $T(h^2w^2c)$, where h, w, and c represent the height of the feature map, the width of the feature map, and the number of channels, respectively. From the above discussions, we have known that time consumption is related to the size of the feature map, so we inserted the attention extraction module after the last convolutional layer. Our method shows comparable time consumption against baselines, where the computational overhead is tiny under both inductive and transductive settings.

6 Conclusion

The interactive attention extraction module based on the traditional feature extraction method has been developed in this paper, which reduces the effect on the bias of the trained model to the target, therefore solving the problem of few-shot classification through meta-learning. Then, the GNN has been utilized as a classification module to fully excavate the relationship between feature vectors and complete the classification at the node-level while improving the accuracy of classification. Experimental results show that the proposed method has a more outstanding performance than most traditional methods of few-shot classification. In future work, the proposed model will be ulteriorly extended to the tasks of zero-shot node classification and few-shot graph-level classification problems as well as the field of text and video classification tasks.

Conflict of interest statement

None declared.

References

Bronstein, M. M., Bruna, J., LeCun, Y., Szlam, A., & Vandergheynst, P. (2017). Geometric deep learning: Going beyond Euclidean data. *IEEE Signal Processing Magazine*, 34(4), 18–42.

Cai, Q., Pan, Y., Yao, T., Yan, C., & Mei, T. (2018). Memory matching networks for one-shot image recognition. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, (pp. 4080–4088).

Cai, X., Zhao, H., Shang, S., Zhou, Y., Deng, W., Chen, H., & Deng, W. (2021). An improved quantum-inspired cooperative coevolution algorithm with multi-strategy and its application. *Expert Systems with Applications*, 171, 114629.

Deng, W., Xu, J., Gao, X.-Z., & Zhao, H. (2020a). An enhanced msiqde algorithm with novel multiple strategies for global optimization problems. *IEEE Transactions on Systems, Man, and Cybernetics: Systems*. doi: 10.1109/TSMC.2020.3030792.

Deng, W., Xu, J., Zhao, H., & Song, Y. (2020b). A novel gate resource allocation method using improved pso-based qea. *IEEE Transactions on Intelligent Transportation Systems*. doi: 10.1109/TITS.2020.3025796.

Deng, W., Shang, S., Cai, X., Zhao, H., Song, Y., & Xu, J. (2021a). An improved differential evolution algorithm and its application in optimization problem. *Soft Computing*, 25(7), 5277–5298.

Deng, W., Xu, J., Song, Y., & Zhao, H. (2021b). Differential evolution algorithm with wavelet basis function and optimal mutation strategy for complex optimization problem. *Applied Soft Computing*, *100*, 106724.

Devlin, J., Chang, M.-W., Lee, K., & Toutanova, K. (2018). Bert: Pre-training of deep bidirectional transformers for language understanding. preprint arXiv:1810.04805.

Ding, K., Wang, J., Li, J., Shu, K., Liu, C., & Liu, H. (2020). Graph prototypical networks for few-shot learning on attributed networks. In *Proceedings of the 29th ACM International Conference on Information and Knowledge Management*, (pp. 295–304).

Finn, C., Abbeel, P., & Levine, S. (2017). Model-agnostic meta-learning for fast adaptation of deep networks. In *International Conference on Machine Learning*, (pp. 1126–1135).

Garcia, V., & Bruna, J. (2017). Few-shot learning with graph neural networks. preprint arXiv:1711.04043.

Gidaris, S., & Komodakis, N. (2019). Generating classification weights with gnn denoising autoencoders for few-shot learning. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, (pp. 21–30).

Guo, Q., Feng, W., Zhou, C., Huang, R., Wan, L., & Wang, S. (2017). Learning dynamic siamese network for visual object tracking. In *Proceedings of the IEEE International Conference on Computer Vision*, (pp. 1763–1771).

Guo, Y., & Cheung, N.-M. (2020). Attentive weights generation for few shot learning via information maximization. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, (pp. 13499–13508).

Jamal, M. A., & Qi, G.-J. (2019). Task agnostic meta-learning for few-shot learning. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, (pp. 11719–11727).

Joachims, T. et al. (1999). Transductive inference for text classification using support vector machines. In *ICMl*(Vol. *99*, pp. 200–209).

Kim, J., Kim, T., Kim, S., & Yoo, C. D. (2019). Edge-labeling graph neural network for few-shot learning. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, (pp. 11–20).

Koch, G., Zemel, R., & Salakhutdinov, R. (2015). Siamese neural networks for one-shot image recognition. In *ICML Deep Learning Workshop*(Vol. *2*).

Lake, B. M., Salakhutdinov, R., & Tenenbaum, J. B. (2015). Human-level concept learning through probabilistic program induction. *Science*, *350*(6266), 1332–1338.

Li, C., Yang, C., Wan, J., Annamalai, A., & Cangelosi, A. (2017). Neural learning and kalman filtering enhanced teaching by demonstration for a baxter robot. In *2017 23rd International Conference on Automation and Computing (ICAC)*, (pp. 1–6). IEEE.

Li, C., Yang, C., & Giannetti, C. (2019a). Segmentation and generalisation for writing skills transfer from humans to robots. *Cognitive Computation and Systems*, *1*(1), 20–25.

Li, W., Wang, L., Xu, J., Huo, J., Gao, Y., & Luo, J. (2019b). Revisiting local descriptor based image-to-class measure for few-shot learning. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, (pp. 7260–7268).

Lin, G., Yang, Y., Fan, Y., Kang, X., Liao, K., & Zhao, F. (2020). High-order structure preserving graph neural network for few-shot learning. preprint arXiv:2005.14415.

Liu, Y., Lee, J., Park, M., Kim, S., Yang, E., Hwang, S. J., & Yang, Y. (2018). Learning to propagate labels: Transductive propagation network for few-shot learning. preprint arXiv:1805.10002.

Luo, C., & Yuille, A. L. (2019). Grouped spatial-temporal aggregation for efficient action recognition. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, (pp. 5512–5521).

Luo, Y., Huang, Z., Zhang, Z., Wang, Z., Baktashmotlagh, M., & Yang, Y. (2020). Learning from the past: Continual meta-learning with Bayesian graph neural networks. In *The Thirty-Fourth AAAI Conference on Artificial Intelligence (AAAI-20)*(Vol. *34*, pp. 5021–5028).

Ma, N., Bu, J., Yang, J., Zhang, Z., Yao, C., Yu, Z., Zhou, S., & Yan, X. (2020). Adaptive-step graph meta-learner for few-shot graph classification. In *Proceedings of the 29th ACM International Conference on Information and Knowledge Management*, (pp. 1055–1064).

Mishra, N., Rohaninejad, M., Chen, X., & Abbeel, P. (2017). A simple neural attentive meta-learner. preprint arXiv:1707.03141.

Munkhdalai, T., & Yu, H. (2017). Meta networks. In *International Conference on Machine Learning*, (pp. 2554–2563).

Nichol, A., Achiam, J., & Schulman, J. (2018). On first-order meta-learning algorithms. preprint arXiv:1803.02999.

Pfahringer, B., Bensusan, H., & Giraud-Carrier, C. G. (2000). Meta-learning by landmarking various learning algorithms. In *ICML*, (pp. 743–750).

Ren, M., Triantafillou, E., Ravi, S., Snell, J., Swersky, K., Tenenbaum, J. B., Larochelle, H., & Zemel, R. S. (2018). Meta-learning for semi-supervised few-shot classification. preprint arXiv:1803.00676.

Santoro, A., Bartunov, S., Botvinick, M., Wierstra, D., & Lillicrap, T. (2016). Meta-learning with memory-augmented neural networks. In *International Conference on Machine Learning*, (pp. 1842–1850).

Silver, D. et al. (2016). Mastering the game of go with deep neural networks and tree search. *Nature*, *529*(7587), 484–489.

Snell, J., Swersky, K., & Zemel, R. S. (2017). Prototypical networks for few-shot learning. preprint arXiv:1703.05175.

Song, Y., Wu, D., Wagdy Mohamed, A., Zhou, X., Zhang, B., & Deng, W. (2021). Enhanced success history adaptive de for parameter optimization of photovoltaic models. *Complexity*, *2021*, 6660115.

Sung, F., Yang, Y., Zhang, L., Xiang, T., Torr, P. H., & Hospedales, T. M. (2018). Learning to compare: Relation network for few-shot learning. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, (pp. 1199–1208).

Triantafillou, E., Zhu, T., Dumoulin, V., Lamblin, P., Evci, U., Xu, K., Goroshin, R., Gelada, C., Swersky, K., Manzagol, P.-A., & Larochelle, H. (2019). Meta-dataset: A dataset of datasets for learning to learn from few examples. preprint arXiv:1903.03096.

Veličković, P., Cucurull, G., Casanova, A., Romero, A., Lio, P., & Bengio, Y. (2017). Graph attention networks. preprint arXiv:1710.10903.

Vinyals, O., Blundell, C., Lillicrap, T., Kavukcuoglu, K., & Wierstra, D. (2016). Matching networks for one shot learning. preprint arXiv:1606.04080.

Wu, Z., Pan, S., Chen, F., Long, G., Zhang, C., & Yu, P. S. (2021). A comprehensive survey on graph neural networks. *IEEE Transactions on Neural Networks and Learning Systems*.*32*(1), pp. 4–24. doi: 10.1109/TNNLS.2020.2978386.

Zhang, Z., Zhao, Y., Liu, J., Wang, S., Tao, R., Xin, R., & Zhang, J. (2019). A general deep learning framework for network reconstruction and dynamics learning. *Applied Network Science*, *4*(1), 1–17.

Zhu, X., & Goldberg, A. B. (2009). Introduction to semi-supervised learning. *Synthesis Lectures on Artificial Intelligence and Machine Learning*, *3*(1), 1–130.