Graph-based few-shot learning with transformed feature propagation and optimal class allocation

Ruiheng Zhang [a], Shuo Yang [b], Qi Zhang [c,*], Lixin Xu [a], Yang He [e], Fan Zhang [d]

[a] School of Mechatronical Engineering, Beijing Institute of Technology, China
[b] Faculty of Engineering and IT, University of Technology Sydney, Australia
[c] School of Civil and Environmental Engineering, University of New South Wales, Australia
[d] College of Information Science and Technology, Beijing University of Chemical Technology, China
[e] High Performance Computing, Agency for Science, Technology and Research, Singapore

A B S T R A C T

Graph neural network has shown impressive ability to capture relations among support(labeled) and query(unlabeled) instances in a few-shot task. It is a feasible way that features are extracted using a pre-trained backbone network, and later adjusted in a few-shot scenario with an episodic meta-trained graph network. However, these adjusted features cannot well represent the few-shot data characteristics owing to the feature distribution mis-match caused by the different optimizations between the backbone and the graph network (*multi-class pre-train* v.s. *episodic meta-train*). Additionally, learning from the limited support instances fails to depict true data distributions thus cause incorrect class allocation. In this paper, we propose to transform the features extracted by a pre-trained self-supervised feature extractor into a Gaussian-like distribution to reduce the feature distribution mis-match, which significantly benefits the later meta-training of the graph network. To tackle the incorrect class allocation, we propose to leverage support and query instances to estimate class centers by computing an optimal class allocation matrix. Extensive experiments on few-shot benchmarks demonstrate that our graph-based few-shot learning pipeline outperforms baseline by 12%, and surpasses state-of-the-art results by a large margin under both full-supervised and semi-supervised settings.

1. Introduction

Humans have the ability to quickly acquire knowledge and generalize them well after seeing only a few examples of them, while it is extremely difficult for machines to do the same [1]. Although deep learning [2,3] has made impressive progress in many computer vision tasks [4–6], the demand for a large number of training data limits its use in many application scenarios [7–10]. Limited training data restrict the generalization of learning-based methods. Reducing the cost of annotating data brings great attention to the research of few-shot learning. To mimic the aforementioned human ability, few-shot learning [11,12], aiming to learn from a few labeled data (support set) and make a good prediction on unlabeled data (query set), attracts considerable attention [13,14].

A practical strategy for few-shot learning is to first pre-train a learning model using a large scale dataset and fine-tune it for a specific task with only small number of training samples. However, this solution may suffer from over-fitting [15] as one or a few instances are insufficient to model the data distributions of the novel(unlabeled) classes. Some research aims to apply data augmentation [16] or mix-up [17] methods to enlarge the training set. Although these techniques may alleviate overfitting in such a limited-data regime, they do not solve it. Meta-learning, also called as learning to learn, intends to leverage previous learning experience to adapt to a new learning scenario rapidly, to address the over-fitting problem explicitly. It samples few-shot learning tasks from a large scale dataset for pre-training, and optimizes the model to perform well on these tasks. Meta-learning methods adopt an episodic training strategy, which randomly samples only a few examples (*e.g.*, 1 or 5) from each class in an episode. By this way, the models are trained episode by episode to gain the ability to learn from few samples.

A rising trend in recent researches was to process the training data with Graph neural network (GNN), which has shown its impressive ability to manipulate structured data. Data instances in an episode can naturally form an undirected acyclic graph, in

* Corresponding author.

which nodes represent data instances and edges represent the similarities between two connected nodes. Thus, few-shot learning can be regarded as the edge prediction of the graph. This formulation is benefit for support set with additional supplementary information, and makes the model more robust. Many research works use graph neural network (GNN) to capture the intra- and inter-class relations in meta-learning, such as Attention-GNN [18], EGNN [19], TPN [20], DPGN [21,22], and Free-DC [23].

Most graph-based few-shot learning methods employ a backbone network to extract and a graph network to propagate features. Some works train both backbone and graph networks in few-shot scenarios solely with episodic strategy, resulting unrepresentative features. Pre-training followed by episodic meta-training has been proven to be a more effective strategy [24]25. It is a feasible way that features are extracted using a pre-trained backbone network, and later adjusted in a few-shot scenario with an episodic meta-trained graph network.

Nevertheless, an optimization problem between the backbone *multi-class pre-training* and the graph network *episodic meta-training* differs a lot. The backbone network is trained over the whole base class data (*e.g.*, 64 classes, 600 samples per class in *mini*ImageNet), while the graph network utilizes episodic meta-training (*e.g.*, 5 classes, 5 samples per class) to learn from a few samples. In [25], the authors prove that the classical pre-training and episodic meta-training produce different feature distributions. The features extracted by the pre-trained backbone network are too complex/mis-match to well suit the graph network few-shot meta-training. Additionally, assigning class labels by limited, sometimes even one, support node per class may cause severe classification incorrection because the limited support nodes are not enough to depict the true data distributions (See Fig. 3). The instances in the query set are unlabeled thus cannot be directly used to computed class centers. Inspired by this, we explore a distribution transport method to make the class center estimation using both support set and query set possible.

In this paper, we attempt to address the issues mentioned above and achieve a robust few-shot visual classification. Firstly, we pre-train the backbone network with self-supervision on base data set to extract features. Then these features are transformed to be aligned with a Gaussian assumption to reduce the distribution mis-match. This feature transformation makes features fit into the considered few-shot classification task and benefit the graph network episodic meta-training in the next step. After the graph feature propagation, we propose to employ support and query nodes to estimate class centers by an optimal transport algorithm to reduce the uncertainty of allocating classes. The illustration of our method is shown in Fig. 1. Addressing two problems that exist in graph-based few-shot learning method, the major contributions of our work can be summarized as follows:

- We are the first to deal with the issue of feature distribution mis-match caused by the different training strategies between the backbone network and the graph network (*multi-class pre-training* v.s. *episodic meta- training*). A feature distribution transformation is proposed to transform the features between backbone and graph network to reduce the distribution mis-match and make distributions more aligned to a Gaussian assumption. The qualitative and quantitative analysis prove the distribution transformation makes the features more suitable for a few-shot scenario and benefits the graph feature propagation.
- Inspired by the optimal transportation problem, we propose to leverage support and query instances to estimate class centers simultaneously by computing an optimal allocation matrix rather than simply assigning query labels with their nearest support labels. Experiments show that the class centers estimated by the proposed optimal class allocation method can better describe the overall data distribution and improve the classification performance.
- Extensive experiments on *mini*ImageNet and *tiered*ImageNet have demonstrated our method achieves a significant improvement in both few-shot classification and semi-supervised few-shot classification tasks.

The remainder of this paper is organized as follows. We review previous progress of meta-learning, graph neural network and optimal transport briefly in Section 2. Then, we present the proposed graph-based few-shot learning method in Section 3. Subsequently, the experiment and ablation studies are performed in Section 4. Finally, we draw a short conclusion and describe some future work in Section 5.

2. Related Work

2.1. Meta-learning

Meta-learning learns meta-level knowledge, *i.e.* knowledge about knowledge, across batches of tasks. The mainstream works of meta-learning can be broadly categorized into three groups: *Optimization-based*, *Generation-based* and *Metric-based*. Optimization-based methods train a well-initialized optimizer to

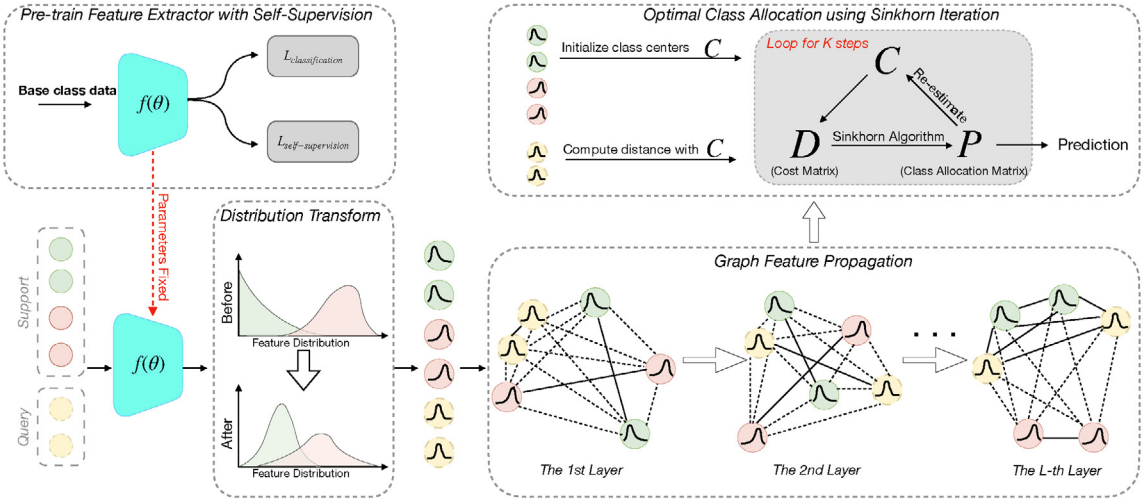

Fig. 1. An overall illustration of our proposed few-shot classification method.

quickly adapt it to unseen classes with a few epochs of training. In [26], the proposed method learned to approximate gradient descent with a LSTM. The proposed method in [27] learned model-agnostic initial parameters and [28] learned the learner update direction and learning rate. Generation-based methods learned to augment data with a generative meta-learner [29], or learned to predict classification weights for novel classes [30,31]. Metric-based meta-learning aims to learn proper distance metrics, such as the matching network [32] and the prototypical network [33].

Many research works use graph neural network (GNN) to capture the intra- and inter-class relations in meta-learning. Specifically, the work [18] proposed to build a complete graph network, where each node feature was concatenated with the corresponding class label, then node features were updated via the attention mechanism of graph network to propagate the label information. Different from this research in [18] which cast the few-shot learning as a node classification problem, EGNN [19] modeled the few-shot learning as an edge labeling problem to predict whether the associated two nodes belong to the same class. In the same vein, the transductive propagation network (TPN) [20] developed a transductive setting to propagate one-hot encoded labels over the entire labeled and unlabeled instances as a whole. DPGN [21,22] constructed a dual complete graph network to combine the distribution-level relations and instance-level relations among all examples.

2.2. Graph Neural Network

Graph neural networks (GNN) were proposed to process graph-structured data [34,35]. It augments the node representation by aggregating and transforming neighboring nodes recursively [36]. A rising trend in recent metric-base meta-learning exploits GNN to capture the intra- and inter-task relations. In [18,37], the authors cast few-shot learning as the node classification problem with GNN. Liu et al. [20] brought the transductive setting into graph-based few-shot learning, which propagated labels from support set to query set in the graph. Kim et al. [19] used the similarity/dissimilarity between samples and dynamically updated both node and edge features for complicated interactions. Yang et al. [21] constructed a dual complete graph to combine the distribution-level and instance-level relations among all examples. RGEN [38] modeled the relations among local image regions with the region-based relation reasoning into zero-shot learning. Lu et al. [39] built an AGNN model to fully capture knowledge from the relational visual data, enabling more accurate object discovery and segmentation. SAGNN [40] proposed an end-to-end scale-aware graph neural network by reasoning the cross-scale relations among the support-query images for few-shot semantic segmentation. Graph-based few-shot learning uses a backbone network to extract and a GNN to propagate example features. The labels of query nodes are assigned with the labels of support nodes connected with them. Some works aforementioned trained both backbone and graph networks in few-shot scenario with an episodic strategy, which weakened the ability of the backbone to extract representative example features. In our method, we firstly pre-train the backbone network on a base class set. Then episodic strategy is applied to train the graph network to adjust features in a few-shot scenario. We also perform feature transformation to bridge the the gap of optimization problem between the pre-trained backbone and the meta-trained graph network.

2.3. Optimal Transport

Optimal transport provides a way to infer the correspondence between two distributions. Recently, it has received great attention in various computer vision tasks. The work of [41] learned an optimal transportation plan from source domain to target domain to solve domain adaption problem. In [42,43], authors employed optimal transport to deal with the 3D shape matching and surface registration problem. The work in [44,45] exploited the effectiveness of optimal transport on graph matching problem [46–48]. To reduce the uncertainty of allocating classes with limited labeled nodes, we compute an optimal class allocation matrix by the optimal transport. This allocates classes for each query node and iteratively update estimated class centers according to the optimal class allocation matrix. In this paper, we tend to leverage optimal transport to handle the incorrect class allocation.

3. Methodology

In this section we introduce the proposed graph-based few-shot learning method, as illustrated in Fig. 1. The algorithm of the proposed method is given in Algorithm 1.

3.1. Problem Definition

Given the training set $\mathcal{D}_{base} = \{(\mathbf{x}_i, y_i)_{i=1}^{m_b}, y_i \in \mathcal{C}_{base}\}$ and testing set $\mathcal{D}_{novel} = \{(\mathbf{x}_i, y_i)_{i=1}^{m_n}, y_i \in \mathcal{C}_{novel}\}$, where m_b and m_n are numbers of samples in $\mathcal{D}_{base}$ and $\mathcal{D}_{novel}$, respectively, $\mathcal{C}_{base}$ and $\mathcal{C}_{novel}$ are class sets of $\mathcal{D}_{base}$ and $\mathcal{D}_{novel}$, respectively, $\mathcal{C}_{base} \cap \mathcal{C}_{novel} = \varnothing$. Few-shot learning aims to learn a model on $\mathcal{D}_{base}$, which is capable of well generalizing the unseen test set $\mathcal{D}_{novel}$ with only a few labeled samples per class. Generally, we can pre-train a classifier over the large-scale base class data $\mathcal{D}_{base}$ then fine-tune the classifier on the $\mathcal{D}_{novel}$. However, a small number of labeled samples from each class are not sufficient to train a model fully reflecting the inter- and intra-class variations.

Episodic training [32] is a training paradigm that mimics the few-shot learning setting of test tasks. Given a large labeled training dataset, episodic training samples a series of training tasks (episodes) $\mathcal{T}$ from a task distribution $P(\mathcal{T})$. In an episodic training, the distribution of training tasks is assumed to be similar to that of test tasks. Specifically, a N-way K-shot classification setting is used for both training and testing stage. Each few-shot task $\mathcal{T}$ has a *support set* $\mathcal{S}$ and a *query set* $\mathcal{Q} : \mathcal{T} = \mathcal{S} \cup \mathcal{Q}$ where $\mathcal{S} = \{(\mathbf{x}_i, y_i)\}_{i=1}^{N\times K}$ and $\mathcal{Q} = \{(\mathbf{x}_i, y_i)\}_{i=N\times K+1}^{N\times K+N\times q}$. The support set $\mathcal{S}$ contains N classes with K samples for each class and the query set $\mathcal{Q}$ has q samples per class. In the training stage, data labels are provided for both support set $\mathcal{S}$ and query set $\mathcal{Q}$, where $\mathcal{S}, \mathcal{Q} \in \mathcal{D}_{base}$. Given testing data $\mathcal{D}_{novel}$, classifier should be able to map the query sample from $\mathcal{Q} \in \mathcal{D}_{novel}$ to the corresponding label with few support samples from $\mathcal{S} \in \mathcal{D}_{novel}$. The label space of training and testing tasks are mutually exclusive $\mathcal{C}_{base} \cap \mathcal{C}_{novel} = \varnothing$.

3.2. Pre-training Feature Extractor with Self-Supervision

Self-supervised learning aims to extract supervisory signals by defining surrogate tasks using only the structural information presented in the data. In the first training stage, we pre-train our feature extractor on the base class data $\mathcal{D}_{base}$ with self-supervision. More specifically, a rotation pretext task is considered in our model to obtain features that can be useful for fast few-shot knowledge transfer. The input images $\mathbf{x} \in \mathcal{D}_{base}$ are rotated by r degrees, $r \in C_R = \{0°, 90°, 180°, 270°\}$. And an auxiliary loss (based on the predicted rotation angle) is added to the standard classification loss. The self-supervision loss is formulated as:

$$\mathcal{L}_{rot} = \frac{1}{|C_R|} * \sum_{\mathbf{x}\in\mathcal{D}_{base}} \sum_{r\in C_R} \mathcal{L}(FC_{rot} \circ f_{emb}(\mathbf{x}^r), r), \tag{1}$$

where $|C_R|$ denotes the cardinality of C_R, $f_{emb}(\mathbf{x}^r)$ is the feature embedding of rotated image $\mathbf{x}^r$, FC_{rot} is the linear classifier for rotation degree prediction, $\mathcal{L}$ is the cross-entropy loss. The image classification loss is given by:

$$\mathcal{L}_{class} = \mathbb{E}_{(\mathbf{x},y)\in\mathcal{D}_{base}, r\in C_R}[\mathcal{L}(FC_{class} \circ f_{emb}(\mathbf{x}^r), y)], \tag{2}$$

where $\mathcal{L}$ is the cross-entropy loss, $f_{emb}(\mathbf{x}^r)$ is the feature embedding of rotated image $\mathbf{x}^r$, FC_{class} is the linear classifier for image classification. The total objective function of the embedding module is the combination of the self-supervision loss and the classification loss:

$$\mathcal{L}_{emb} = \mathcal{L}_{class} + \mathcal{L}_{rot}. \tag{3}$$

After the self-supervised pre-training stage, the feature extractor parameters are determined to extract image features used for the later episodic meta-training of the graph propagation network.

3.3. Feature Distribution Transform

Many recent works proved that pre-training + meta-training is a better strategy than solely episodic meta-training [24,49]. We also found that pre-training the backbone then meta-training the graph network brings better performance in graph-based few-shot learning (Table 4). However, the pre-trained backbone f_{emb} is optimized on the large-scale diverse and fully annotated base class data D_{base}, which differs a lot to the considered few-shot tasks. Additionally, the classical pre-training and episodic meta-training produce different feature distributions [25]. Therefore, we apply a feature distribution transformation on all features extracted by backbone to make them more aligned to a Gaussian distribution before feeding them to the episodic meta-training of graph network.

Empirically, many previous works [23] found that the image features extracted by a convolutional neural network form Gaussian distribution well. From the intuitive perspective, the features from the same class tend to cluster well. The center of the cluster can be regarded as the class mean. Thus we leverage a power transformation to transform the feature distribution and empirically found it works very well.

Note that image embedding features are extracted by pre-trained extractor as $\mathbf{f} = f_{emb}(\mathbf{x})$. The following power transformation is performed to make the distribution of $\mathbf{f}$ more Gaussian:

$$\begin{cases} \widetilde{\mathbf{f}} = \frac{(\mathbf{f}+\epsilon)^\beta}{\|(\mathbf{f}+\epsilon)^\beta\|_2} & \text{if } \beta \neq 0, \\ \widetilde{\mathbf{f}} = \frac{\log(\mathbf{f}+\epsilon)}{\|\log(\mathbf{f}+\epsilon)\|_2} & \text{if } \beta = 0. \end{cases} \tag{4}$$

Here, ϵ equals to $1e-6$ to make $(\mathbf{f}+\epsilon)$ strictly positive. β is a hyper-parameter to adjust the feature distribution. The above transformation can reduce the mis-match of distribution thus make it as a Gaussian-like distribution. We prove that the feature distribution transformation can reduce the distribution mis-match and benefit the episodic training of graph network qualitatively and quantitatively in the experiments section (Table 4 and Fig. 2).

3.4. Graph Feature Propagation

Few-shot learning in an episode needs to fully exploit the relationships between support set $\mathcal{S}$ and query set $\mathcal{Q}$. Therefore using graph neural networks to iteratively gather neighbor feature information can express complex interactions among data instances. As shown in Fig. 1, the support data $\mathcal{S} = \left\{\left(\widetilde{\mathbf{f}}_i, y_i\right)\right\}_{i=1}^{N\times K}$ and query data $\mathcal{Q} = \left\{\left(\widetilde{\mathbf{f}}_i, y_i\right)\right\}_{i=N\times K+1}^{N\times K+N\times q}$ in each episode $\mathcal{T} = (\mathcal{S}, \mathcal{Q})$ can form an undirected acyclic graph $\mathcal{G} = (\mathcal{V}, \mathcal{E})$. Each node $v_i \in \mathcal{V}$ represents an image embedding vector and each edge $e_{ij} \in \mathcal{E}$ represents an interaction between nodes. The adjacency matrix $\mathcal{A} = \{A_{ij}\}_{i,j=1}^{|\mathcal{V}|}$ is defined as semantic similarity between nodes v_i and v_j, which is dynamically updated along with the propagation. Given the support set label, the adjacency matrix is initialized as:

$$A_{ij}^{(0)} = \begin{cases} 1 & \text{if } \widetilde{\mathbf{f}}_i, \widetilde{\mathbf{f}}_j \in \mathcal{S} \text{ and } y_i = y_j, \\ 0 & \text{if } \widetilde{\mathbf{f}}_i, \widetilde{\mathbf{f}}_j \in \mathcal{S} \text{ and } y_i \neq y_j, \\ 0.5 & \text{otherwise.} \end{cases} \tag{5}$$

We construct a L-layers graph to propagate the embeddings to gather information from neighbors L hops away. The forward propagation of the graph network is an alternate update of node features and adjacency matrix (edge features) through layers.

3.4.1. Node Feature Aggregation

Given $v_i^{(\ell-1)}$ and $A_{ij}^{(\ell-1)}$ from the layer $\ell-1$, node features are firstly updated by neighbor information aggregation. The node feature v_i^ℓ at the layer ℓ is updated by aggregating neighbor node features according to their similarity $A_{ij}^{(\ell-1)}$:

$$\hat{v}_i^{(\ell-1)} = \sum_j \left(v_i^{(\ell-1)} A_{ij}^{(\ell-1)}\right), \tag{6}$$

$$v_i^{(\ell)} = f_{node}\left(\left[v_i^{(\ell-1)}; \hat{v}_i^{(\ell-1)}\right]\right), \tag{7}$$

where f_{node} is a node feature transformation network and $[\cdot;\cdot]$ is the concatenation operation. The node features at the 0^{th}-layer are initialized with the extracted features with distribution transformation, *i.e.*, $v_i^{(0)} = \widetilde{\mathbf{f}}_i$.

3.4.2. Adjacency Matrix Update

After obtaining node features at the ℓ^{th} layer, the adjacency features at the ℓ^{th} layer $A_{ij}^{(\ell)}$ can be updated by:

$$\widetilde{A}_{ij}^{(\ell)} = f_{edge}\left(\left\|v_i^{(\ell)} - v_j^{(\ell)}\right\|\right), \tag{8}$$

$$A_{ij}^{(\ell)} = D^{-\frac{1}{2}} \widetilde{A}_{ij}^{(\ell)} D^{-\frac{1}{2}}, \tag{9}$$

where D is the degree matrix of adjacency matrix, f_{edge} is the edge feature transformation network.

3.4.3. Training

The parameters of the node feature transformation network f_{node} and the edge feature transformation network f_{edge} are trained by minimizing the binary cross-entropy loss of ground truth query edge-labels and the predicted adjacency matrix $\mathcal{A}$:

$$\mathcal{L}_e = \sum_{\ell=1}^{L} \lambda_\ell \mathcal{L}\left(Y_{edge}, \mathcal{A}^{(\ell)}\right), \tag{10}$$

where Y_{edge} is the set of all ground-truth edge-labels and $\mathcal{A}^{(\ell)}$ is the predicted adjacency matrix at the ℓ^{th} layer. λ_ℓ is hyper-parameter. Notably, in the training phase, we hold the ground truth of support and query training set to form a similarly matrix, as the supervisor signal. This matrix is generated by support set label and query set label, as Y_{edge} and $\mathcal{A}^{(\ell)}$.

3.5. Optimal Class Allocation using Sinkhorn Iteration

After the graph propagation, query nodes are closer to their corresponding support nodes from the same class. Therefore we can simply assign the query labels with their closest class centers computed by averaging all support node features in each class. This works well when labeled support data are sufficient. However, estimating class centers using very limited labeled data (1 or 5)

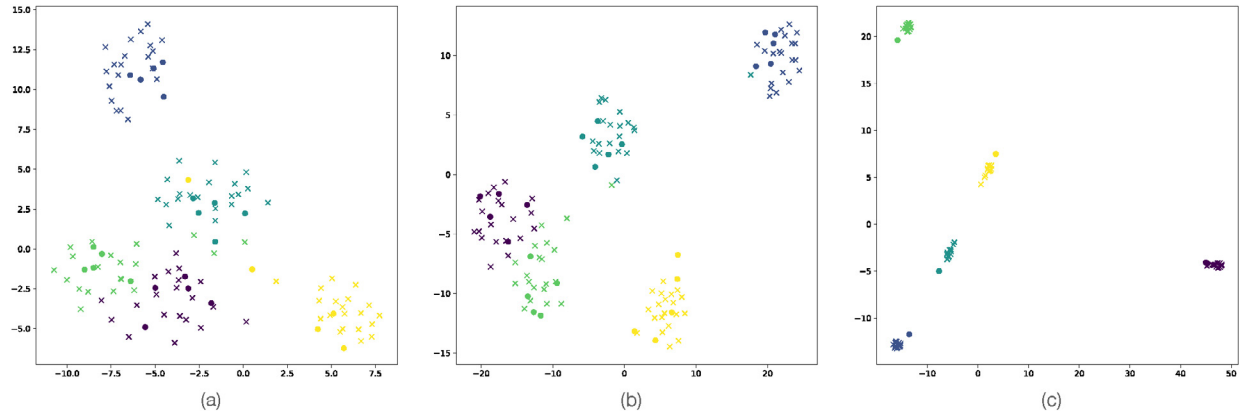

Fig. 2. t-SNE visualization of features in a 5way-5shot experiment. (a) Features before distribution transformation. (b) Features after distribution transformation. (c) Features after graph propagation. 'o' represents support, 'x' represents query. Different colors means different labels.

cannot correctly depict the distribution of the entire class thus cause severe classification incorrection (See Fig. 3). The reason for estimating class center using both support and query set is that, due to the extremely limited number of labeled examples in the support set, e.g., 1 or 5, the support set's mean is extremely biased and far from mirroring the ground-truth class center. Using both support set and query set to estimate the class center can greatly calibrate the biased class center and result in a more representative class center feature.

Optimal transport aims at computing minimal cost transportation from a source distribution to a target distribution. Inspired by optimal transport algorithm [50], we propose to estimate class centers by computing optimal transportation between query nodes and class centers. Denoting support and query nodes at the last layer of the graph network as $\mathcal{V}_S$ and $\mathcal{V}_Q$, respectively. We firstly initialize N class centers by averaging labeled support nodes in each class:

$$\mathbf{c}_j = \frac{1}{K} \cdot \sum_{v \in \mathcal{V}_S, v \in \mathcal{C}_j} v, \quad j = 1 \dots N, \tag{11}$$

where N and K are numbers of classes and samples per class, respectively, *i.e.*, N-way K-shot mentioned above. $\mathcal{C}_j$ is class j. Then, we define a cost matrix $\mathbf{D}_{ij}$ which denotes the euclidean distance between query node v_i and class center $\mathbf{c}_j$. After that, we use sinkhorn algorithm to compute a class allocation matrix to allocate unlabeled query samples to class centers with a minimal transportation cost:

$$\begin{aligned} \mathbf{P}^* &= Sinkhorn(\mathbf{D}, \lambda) \\ &= \arg\min_{\mathbf{P} \in \mathbb{U}} \sum_{ij} \mathbf{P}_{ij} \mathbf{D}_{ij} + \lambda H(\mathbf{P}), \end{aligned} \tag{12}$$

$H(\mathbf{P})$ denotes the entropy of P regularized by λ, which aims to make the computation practical and effective. $\mathbb{U}$ is a set contains all possible allocation matrices:

$$\mathbb{U} = \left\{ \mathbf{P} \in \mathbb{R}_+^{Nq \times N} | \mathbf{P}\mathbf{1}_N = \mathbf{1}_{Nq}, \mathbf{P}^T \mathbf{1}_{Nq} = q\mathbf{1}_N \right\}, \tag{13}$$

where N is the number of classes and q is the number of query samples per class. The $\mathbf{P}\mathbf{1}_N = \mathbf{1}_{Nq}$ and $\mathbf{P}^T\mathbf{1}_{Nq} = q\mathbf{1}_N$ are constraints of allocation matrix [50]. The value of $\mathbf{P}_{ij}$ means the probability of allocating query sample v_i to class j. The sinkhorn algorithm provides an efficient method to compute the optimal allocation matrix that can minimize the transportation cost.

As formulated in Eq. (11), we initialize the class center $\mathbf{c}_j$ by averaging support node features. To use unlabeled query nodes in the class center estimation, we alternately update class centers and class allocation matrix. At each step, an allocation matrix $\mathbf{P}$ on the unlabeled query nodes is computed. Then $\mathbf{c}_j$ is re-estimated along with labeled support nodes:

$$\boldsymbol{\mu}_j = \frac{\sum_{i=1}^{Nq} \mathbf{P}_{ij}^* v_i + \sum_{v \in \mathcal{V}_S, v \in \mathcal{C}_j} v}{K + \sum_{i=1}^{Nq} \mathbf{P}_{ij}^*}, \tag{14}$$

$$\mathbf{c}_j \leftarrow \mathbf{c}_j + \alpha\left(\boldsymbol{\mu}_j - \mathbf{c}_j\right), \tag{15}$$

where α is a hyper-parameter. After k steps of iterative updates on the allocation matrix $\mathbf{P}^*$, the prediction of sample i can be given as:

$$\hat{y}_i = \arg\max_j \mathbf{P}_{ij}^*. \tag{16}$$

Algorithm 1: Inference process of the proposed method.

Require:
$\mathcal{G} = (\mathcal{V}, \mathcal{E}; \mathcal{T})$, where $\mathcal{T} = \mathcal{S} \bigcup \mathcal{Q}$,
$\mathcal{S} = \{(\mathbf{x}_i, y_i)\}_{i=1}^{N \times K}$, $\mathcal{Q} = \{\mathbf{x}_i\}_{i=N\times K+1}^{N \times K + N \times q}$;
Input:
Node feature transformation network: f_{node};
Edge feature transformation network: f_{edge};
Output:
$\{\hat{y}_i\}_{i=N\times K+1}^{N \times K + N \times q}$;
1: Extract features for input images: $\mathbf{f}_i = f_{emb}(\mathbf{x}_i), \forall i$.
2: Transform feature distributions with Eq. (4);
3: Initialize graph nodes with transformed image features:
$v_i^{(0)} = \widetilde{\mathbf{f}}_i$;
4: Update node representations with Eq. (6) and (7);
5: Compute adjacency matrix with Eq. (8) and (9);
6: Initialize class centers with Eq. (11);
7: **for all** k steps **do**
8: Compute optimal class allocation matrix $\mathbf{P}$ with Eq. (12);
9: Update class centers with Eq. (14) and Eq. (15);
10: **end for**
11: **Return** $\hat{y}_i = \arg\max_j \mathbf{P}_{ij}^*$;

4. Experiments and Discussions

In this section, we firstly introduce the experimental setup, including implementation details and datasets. After that, we compare our results with state-of-the-art algorithms, followed by performing an ablation study.

4.1. Implementation Details

4.1.1. Network Architecture

To make a fair comparison, we use the same embedding networks as the most existing works, such as Conv4, WRN, and ResNet-18. We also adopt the same graph network architecture as [19], which consists of four convolutional blocks including a 3×3 convolutions, a batch normalization, a 2×2 max-pooling and a LeakyReLU activation.

4.1.2. Parameter Settings.

The parameters of different modules are discussed as follows.

Feature extractor Module: We use three popular networks for fair comparison, which are Conv4, ResNet18 and WRN. Conv4 mainly consists of four Conv-BN-ReLU blocks [19]. The last two blocks also contain a dropout layer. ResNet18 is the same as the one described in [51]. It mainly has four blocks, which include two residual blocks for ResNet18. WRN was firstly proposed in [52]. It mainly has three residual blocks and the depth of the network is set to 28 as in [53]. Backbones Conv4, ResNet-18 and WRN are trained by Adam optimizer with an initial learning rate $lr = 1e-3$. We decay the learning rate by 0.1 per 15000 iterations and set the weight decay to $1e-5$. We train the feature extractor on whole base class data (training set) not using episodic strategy. Following the works in [21,17], the output feature dimensions of Conv4, ResNet-18 and WRN are 128, 512 and 640, respectively.

Feature Distribution Transformation Module: In Equation. (4), ϵ equals to $1e-6$ to make $\mathbf{f}+\epsilon$ strictly positive. β is used to control the mis-match of the distribution. We choose $\beta = 0.5$ for the considered datasets.

Graph Feature Propagation Module: At the episodic training stage, our experiments on all datasets are performed using $N = 5, K = 1$ or $N = 5, K = 5$, *i.e.* 5*way*-1*shot* and 5*way*-5*shot*. The number of query samples q is 15, which means we randomly sample 15 queries for each of 5 classes. We use a transductive setting in the previous works [20,19] to process all the query samples simultaneously. The mini-batch size is 80 and 64 for 1-shot and 5-shot experiments, respectively. The number of graph layers is 3. The loss coefficients in Eq. (10) are 0.5, 0.5 and 1, respectively. The graph network is trained by Adam optimizer with an initial learning rate $1e-3$ and weight decay of $1e-6$. The reported results are obtained by averaging classification accuracy over 10000 tasks.

Optimal Class Allocation Module: The coefficient λ of the regularization item in Eq. (12) is chosen to be 10. The learning rate in Eq.(15) and iteration steps of class center update are $\alpha = 0.2$ and $k = 20$, respectively.

4.2. Datasets

We conduct extensive experiments on two widely-used few-shot image classification benchmarks, *i.e. mini*ImageNet, *tiered*ImageNet. Noted that the training set and testing set are also called base class data and novel class data, respectively.

- *mini***ImageNet** is derived from ILSVRC-12 dataset [54]. It contains 100 different classes with 600 samples per class. The image size is $84 \times 84 \times 3$. We follow the splits used in [26], where 64, 16, and 20 classes are used for training, validation, and testing, respectively.
- *tiered***ImageNet** is a larger subset of ILSVRC-12 dataset [54], which contains 608 classes sampled from hierarchical category structure. Each class belongs to one of 34 higher-level categories sampled from the high-level nodes in the ImageNet. The average number of images in each class is 1281. We use 351, 97, and 160 classes for training, validation, and testing, respectively.

4.3. Compared Methods

We compare our method with the following methods:

MAML (Model-Agnostic Meta-Learning) [27] is a classic meta-learning algorithm proposed by C. Finn. This algorithm aims to find an initial parameters of a learning model [55], so that the model can learn quickly on a small amount of training data of new tasks.

MatchingNet (Matching Networks) [32] is an attention- and memory-based network with metric learning developed by Oriol Vinyals. This method needs the environment for testing and training must match.

ProtoNet (Prototypical Networks) [33] regards a few-shot classification task as finding the prototypical center of each class in semantic space [56]. It differs form MatchingNet in metric function.

GNN (Graph Neural Networks) [18] defines a graph neural network architecture that generalizes several of existing few-shot learning models, by assimilating generic message-passing inference algorithms [57] with their neural-network counterparts.

EGNN (Edge-Labeling Graph Neural Network) [19] learns the edge information, that is, the similarities and differences between nodes, through the graph network for information dissemination,

Baseline++ [58] is a variation of the baseline method that follows the standard transfer learning procedure. Baseline++ explicitly reduces intra-class variation among features during training.

LEO (Transductive Propagation Network) [59] is a few-shot learning algorithm based on parameter optimization, by building a latent embedding space.

TPN (Transductive Propagation Network) [20] proposes to use transductive information to propagate the label of training set and test set in one episode, so that the pseudo of unlabeled data could be predicted.

S2M2$_R$ (Self-Supervised Manifold Mix) [17] learns relevant feature manifold using self-supervision and regularization techniques to significantly improve few-shot learning performance.

DPGN (Distribution Propagation Graph Network) [21] is a dual graph neural network model, composing of the Propagation Graph and the Distribution Graph, which fuses the relationship between instance level and distribution level.

LR + ICI(Tran.) (Instance Credibility Inference) [60] obtains the pseudo of unlabeled samples by self-training method, and selects the pseudo with the highest confidence as for data augmentation [61].

4.4. Comparison with the State-of-The-Art

We compare the performance of our method with aforementioned state-of-the-art models on two benchmarks, *mini*ImageNet and *tiered*ImageNet. Among them, EGNN [19] is mostly related to our approach and could be regarded as a baseline of ours.

Performance on *mini*ImageNet: Table 1 reports the results of different method. We can observe that with ConvNet4 as the backbone, our algorithm brings a significant improvement over baseline EGNN [19], *i.e.*, 12% improvement in 1-shot scenario and 7% improvement in 5-shot scenario. Also, our method is superior to all methods with the same backbone ConvNet4. Notably, ours with ConvNet4 outperforms all other method with ResNet18 or WRN in 1-shot scenario. Compared with DPGN [21], ours makes a huge improvement from 66.01% to 71.82 with Conv4 in 1-shot scenario, but makes a dent in 5-shot scenario up by less than 1%. This phenomenon illustrates that our method performs better in 5-way 1-

Table 1
The 5-way 1-shot and 5-shot classification accuracies (%) on the *mini*ImageNet dataset, with 95% confidence interval.

Models	Backbone	1-shot ↑	5-shot ↑
MatchingNet [32]	Conv4	43.56 ± 0.84	55.31 ± 0.73
ProtoNet [33]	Conv4	49.42 ± 0.79	68.20 ± 0.66
MAML [27]	Conv4	48.70 ± 1.84	55.31 ± 0.73
GNN [18]	Conv4	50.33 ± 0.36	66.41 ±0.63
EGNN [19]	Conv4	59.63 ± 0.52	76.34 ± 0.48
DPGN [21]	Conv4	66.01 ± 0.36	82.83 ± 0.41
Ours	**Conv4**	**71.82 ± 0.88**	**83.04 ± 0.51**
Baseline++ [58]	ResNet18	51.87 ± 0.77	75.68 ± 0.63
MAML [27]	ResNet18	49.61 ± 0.92	65.72 ± 0.77
DPGN [21]	ResNet18	66.63 ± 0.51	84.07 ± 0.42
LR + ICI(Tran.) [60]	ResNet12	66.80	79.26
Ours	**ResNet18**	**73.36 ± 0.73**	**85.01 ± 0.91**
LEO [59]	WRN	61.76 ± 0.08	77.59 ± 0.12
$S2M2_R$ [17]	WRN	64.93 ± 0.18	83.18 ± 0.11
DPGN [21]	WRN	67.24 ± 0.51	83.72 ± 0.44
Ours	**WRN**	**74.27 ± 0.33**	**85.13 ± 0.11**

shot settings than in 5-way 5-shot settings. The same trend happens with deeper ResNet18 and WRN. In both 5-way 1-shot and 5-way 5-shot settings, our method with Conv4, ResNet18 and WRN outperforms others.

Performance on *tiered*ImageNet: From Table 2, as can be seen, with ConvNet4 as the backbone, our method brings about 6% and 7% improvement over our baseline EGNN [19] in 1-shot and 5-shot scenario respectively. With the backbone Conv4, our method performs better on *tiered*ImageNet than on *mini*ImageNet. The improvement of ours on *tiered*ImageNet is larger than on *mini*ImageNet compared with DPGN, by 8% and 2%. Besides, the performance of ours is superior to other methods with the same backbone as well. Significantly, on this more challenging dataset, our method with WRN as the backbone achieves the state-of-the-art performance 83.88% in 1-shot scenario and 90.39% in 5-shot scenario, demonstrating the superiority of our method again.

In summary, as shown in Table 1 and Table 2, our method is superior to other existing methods and achieves the state-of-the-art performance equipped with different backbones. A deeper backbone network can usually extract more powerful features. We further equip backbones with self-supervision which will make the features more robust to get better classification performance. We observe that our model especially performs better in 5way-1shot settings. Because previous works using only one labeled support sample to represent the center of the data distribution which cannot depict the real data distribution. On the contrary, our method learns all the data including labeled and unlabeled to estimate class centers, hence can estimate class centers well even with one labeled sample.

Table 2
The 5-way 1-shot and 5-shot classification accuracies (%) on the *tiered*ImageNet dataset, with 95% confidence interval. * indicates re-implementation.

Models	Backbone	1-shot ↑	5-shot ↑
MAML* [27]	Conv4	51.67 ± 1.81	70.30 ± 1.75
ProtoNet* [33]	Conv4	53.34 ± 0.89	72.69 ± 0.74
TPN [20]	Conv4	59.91 ± 0.94	73.30 ± 0.75
EGNN [19]	Conv4	63.52 ± 0.52	80.24 ± 0.49
DPGN [21]	Conv4	69.43 ± 0.49	85.92 ± 0.42
Ours	**Conv4**	**77.67 ± 0.27**	**87.98 ± 1.01**
DPGN [21]	ResNet18	70.46 ± 0.52	86.44 ± 0.41
LR + ICI(Tran.) [60]	ResNet12	80.79	87.92
Ours	**ResNet18**	**83.71 ± 0.50**	**89.35 ± 0.87**
LEO [59]	WRN	66.33 ± 0.05	81.44 ± 0.09
Ours	**WRN**	**83.88 ± 0.25**	**90.39 ± 0.22**

4.5. Performance on Semi-supervised Learning

We follow the setting as [18,19] in our semi-supervised classification experiment. Specifically, the support data is partially labeled in the semi-supervised regime. Table 3 shows the semi-supervised 5way-5shot classification results on *mini*ImageNet. '20%-labeled' means only 20% of support data are labeled while 80% of support data are unlabeled. 'LabeledOnly' denotes learning with only labeled support samples without considering unlabeled support data. 'Semi' means the semi-supervised learning. By leveraging unlabeled samples to estimate class centers, our method reduces the uncertainty of estimating class centers using limited labeled data and achieves the best performance on semi-supervised learning.

4.6. Ablation Study

4.6.1. Role of Each Module

Table 4 shows the importance of different modules in our model. Noted that the baseline method (EGNN) trains both backbone and graph networks in few-shot scenario solely with episodic strategy, which may weaken the ability of the backbone to extract representative features. As we can see, DPGN is more effective than EGNN. We also believe DPGN will reasonably improve our pipeline. Compared with EGNN, our Self-supervised Pre-training (SSP) of backbone network is an effective way to extract representative features and can enhances the generalization and robustness of the extracted features, then graph network is used to adjust those features in a few-shot scenario. Moreover, distribution transformation (DT) can reduce the mis-match of the feature distribution thus bridge the optimization gap between the backbone and the graph network and benefit the graph episodic meta-training. Optimal class allocation (OCA) can reduce the uncertainty of class center estimation by leveraging all the labeled and unlabeled data to estimate the class centers simultaneously using optimal transport algorithm. In Table 4, our method without OCA indicates that we compute class centers by averaging labeled samples only. Fig. 2 shows the t-SNE visualizations of image features in a 5way-5shot episode. The original distribution of features extracted by pre-

Table 3
Semi-supervised few-shot classification accuracies (%) on *mini*ImageNet with 95% confidence intervals. * indicates re-implementation. All models use Conv4 as backbone.

Methods	5way-5shot ↑		
2-4	20%-labeled	40%-labeled	100%-labeled
GNN-LabeledOnly	50.33 ± 0.36	56.91 ± 0.42	66.41 ± 0.63
GNN-Semi	52.45 ± 0.88	58.76 ± 0.86	66.41 ± 0.63
EGNN-LabeledOnly*	58.65 ± 0.55	56.91 ± 0.00	75.25 ± 0.49
EGNN-Semi*	63.62 ± 0.00	64.32 ± 0.00	75.25 ± 0.49
Ours-LabeledOnly	69.56 ± 0.32	75.28 ± 0.05	**83.04 ± 0.51**
Ours-Semi	**73.97 ± 0.73**	**78.75 ± 0.23**	**83.04 ± 0.51**

Table 4
Ablation study on *mini*ImageNet, with 95% confidence interval. * indicates re-implementation. All models use Conv4 as backbone.

Methods	Backbone	*mini*ImageNet (%)	
		1-shot ↑	5-shot ↑
Baseline(EGNN)	Conv4	59.63 ± 0.52	76.34 ± 0.48
Baseline(DPGN*)	Conv4	61.32 ± 0.62	78.29 ± 0.50
Baseline(EGNN)+SSP	Conv4	62.93 ± 0.77	78.37 ± 0.13
Baseline(EGNN)+SSP + DT	Conv4	69.92 ± 0.26	81.31 ± 0.18
Baseline(EGNN)+SSP + DT + OCA (Full)	Conv4	**71.82 ± 0.88**	**83.04 ± 0.51**

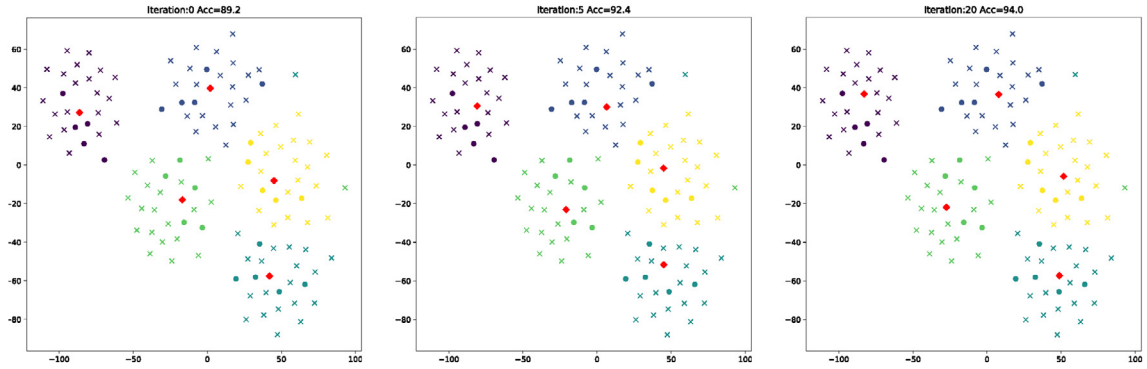

Fig. 3. The visualization of the optimal class allocation process. 'o' represents support, 'x' represents query, red squares represent estimated class centers. The class centers in the left figure (iteration 0) were initialized by support features, which in the middle and right figures were re-estimated by support and query features. The classification accuracies are shown above the figures.

trained feature extractor is shown in Fig. 2 (a), where samples from different classes are separated coarsely but not clearly due to the distribution mis-match. The feature distribution transformation module reduces mis-match of the distributions and makes feature distributions more aligned to the Gaussian assumption in Fig. 2 (b). The graph feature propagation further cluster samples from the same class and pull samples from different classes away in Fig. 2 (c). Our proposed method outperforms the baseline by 12% relatively.

4.6.2. Effect of Optimal Class Allocation

Our optimal class allocation module uses labeled and unlabeled samples to estimate class centers simultaneously by an optimal transportation algorithm. Fig. 3 shows the change of the class centers along with the sinkhorn iteration. Class centers in iteration 0 (left in Fig. 3) are initialized by averaging support samples only. The initialized centers can only represent the centers of support features but not the centers of the whole feature distributions. With our optimal class allocation, the class centers are re-estimated steps by steps by considering the support and query data simultaneously. After 20 iterations, the re-estimated class centers (right in Fig. 3) can represent the centers of the whole distribution thus improve the classification performance.

4.6.3. Parameter Sensitivity Analysis on β

In Eq. 4, ϵ equals to $1e-6$ to make $\mathbf{f}+\epsilon$ strictly positive. β is used to control the mis-match of the distribution. We conduct the parameter sensitivity analysis on β to show the reason why we finally choose $\beta = 0.5$ for the considered datasets. Fig. 4 illustrates the test performance reaches the peak when $\beta = 0.5$.

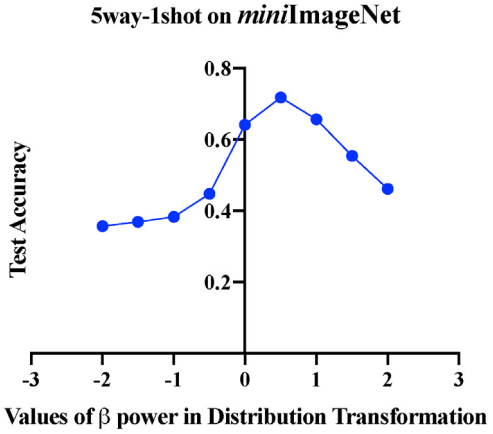

Fig. 4. Parameter sensitivity analysis for β on *mini*ImageNet.

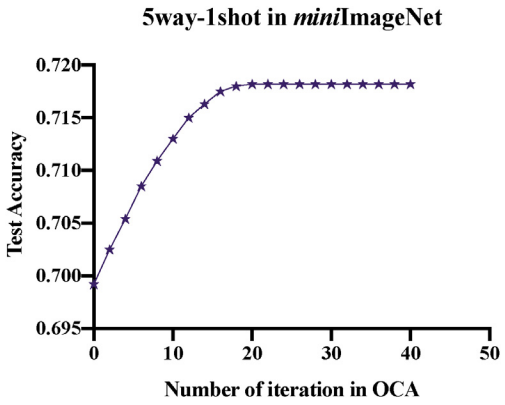

Fig. 5. Accuracy when increasing iteration in OCA on *mini*ImageNet.

4.6.4. Parameter Sensitivity Analysis on Loops of OCA

It is necessary to give evidence of selecting 20 iterations in OCA. Here, we draw the curve of Test Accuracy(5way-1shot) and iteration in OCA on *mini*ImageNet, to help readers have a comprehensive understanding on this hyper-parameter. In Fig. 5, we could observe that accuracy and iteration show positive correlation from iteration = 0 to iteration = 20. In another word, when iterations increases gradually, test accuracy on *mini*ImageNet also climbs up. Once the iteration comes to 20, the accuracy reaches the top, 0.7182. When continuing to increase iterations, the accuracy is almost stable. Considering the Occam's Razor principle, therefore, we select iteration = 20 in OCA.

5. Conclusion

In this paper, we addressed two major problems that exist in the graph-based few-shot learning method. A feature distribution transformation is proposed to reduce the distribution mis-match thus bridge the optimization gap between the backbone network *multi-class pre-training* and the graph network *episodic meta-training*. In order to reduce the uncertainty of allocating classes by limited support nodes, we proposed to leverage support and query nodes to allocate classes simultaneously by computing an optimal class allocation matrix. Experimental results show the significant superiority of our method and prove the effectiveness of each module in our method. Our method can also be well generalized in other graph-based few-shot learning methods.

CRediT authorship contribution statement

Ruiheng Zhang: Conceptualization, Methodology, Writing-original-draft. **Shuo Yang:** Data-curation, Writing-original-draft. **Qi Zhang:** Visualization, Investigation, Supervision, Writing-review-editing. **Lixin Xu:** Supervision. **Yang He:** Software, Validation. **Fan Zhang:** Writing-review-editing.

Declaration of Competing Interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

References

[1] W. Wang, J. Shen, Deep visual attention prediction, IEEE Transactions on Image Processing 27 (5) (2017) 2368–2378.
[2] D. Ravı, C. Wong, F. Deligianni, M. Berthelot, J. Andreu-Perez, B. Lo, G.-Z. Yang, Deep learning for health informatics, IEEE journal of biomedical and health informatics 21 (1) (2016) 4–21.
[3] W. Wang, X. Lu, J. Shen, D.J. Crandall, L. Shao, Zero-shot video object segmentation via attentive graph neural networks, in: Proceedings of the IEEE/CVF International Conference on Computer Vision, 2019, pp. 9236–9245.
[4] W. Wang, J. Shen, L. Shao, Video salient object detection via fully convolutional networks, IEEE Transactions on Image Processing 27 (1) (2017) 38–49.
[5] S. Yang, P. Sun, Y. Jiang, X. Xia, R. Zhang, Z. Yuan, C. Wang, P. Luo, M. Xu, Objects in semantic topology, arXiv preprint arXiv:2110.02687.
[6] S. Yang, W. Yu, Y. Zheng, H. Yao, T. Mei, Adaptive semantic-visual tree for hierarchical embeddings, Proceedings of the 27th ACM International Conference on Multimedia doi:10.1145/3343031.3350995.
[7] A. Iosifidis, A. Tefas, I. Pitas, Approximate kernel extreme learning machine for large scale data classification, Neurocomputing 219 (2017) 210–220.
[8] X. Lu, W. Wang, M. Danelljan, T. Zhou, J. Shen, L. Van Gool, Video object segmentation with episodic graph memory networks, in: Computer Vision–ECCV 2020: 16th European Conference, Glasgow, UK, August 23–28, 2020, Proceedings, Part III 16, Springer, 2020, pp. 661–679.
[9] W. Wang, T. Zhou, S. Qi, J. Shen, S.-C. Zhu, Hierarchical human semantic parsing with comprehensive part-relation modeling, IEEE Transactions on Pattern Analysis and Machine Intelligence.
[10] T. Zhou, S. Qi, W. Wang, J. Shen, S.-C. Zhu, Cascaded parsing of human-object interaction recognition, IEEE Transactions on Pattern Analysis and Machine Intelligence.
[11] H. Song, M.T. Torres, E. Özcan, I. Triguero, L2ae-d: Learning to aggregate embeddings for few-shot learning with meta-level dropout, Neurocomputing 442 (2021) 200–208.
[12] X. Li, Z. Sun, J.-H. Xue, Z. Ma, A concise review of recent few-shot meta-learning methods, arXiv preprint arXiv:2005.10953.
[13] W. Liu, Z. Wang, X. Liu, N. Zeng, Y. Liu, F.E. Alsaadi, A survey of deep neural network architectures and their applications, Neurocomputing 234 (2017) 11–26.
[14] S. Yang, S. Wu, T. Liu, M. Xu, Bridging the gap between few-shot and many-shot learning via distribution calibration doi:10.36227/techrxiv.14380697.v1. url:https://www.techrxiv.org/articles/preprint/Bridging_the_Gap_between_Few-Shot_and_Many-Shot_Learning_via_Distribution_Calibration/14380697.
[15] S. Gidaris, N. Komodakis, Dynamic few-shot visual learning without forgetting, CVPR (2018).
[16] B. Hariharan, R. Girshick, Low-shot visual recognition by shrinking and hallucinating features, in: ICCV, 2017.
[17] P. Mangla, N. Kumari, A. Sinha, M. Singh, B. Krishnamurthy, V.N. Balasubramanian, Charting the right manifold: Manifold mixup for few-shot learning, WACV (2020).
[18] V.G. Satorras, J.B. Estrach, Few-shot learning with graph neural networks, in: International Conference on Learning Representations, 2018. url:https://openreview.net/forum?id=BJj6qGbRW.
[19] J. Kim, T. Kim, S. Kim, C.D. Yoo, Edge-labeling graph neural network for few-shot learning, in: CVPR, 2019.
[20] Y. Liu, J. Lee, M. Park, S. Kim, E. Yang, S. Hwang, Y. Yang, Learning to propagate labels: Transductive propagation network for few-shot learning, in: ICLR, 2019.
[21] L. Yang, L. Li, Z. Zhang, X. Zhou, E. Zhou, Y. Liu, Dpgn: Distribution propagation graph network for few-shot learning, in: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), 2020.
[22] R. Zhang, L. Xu, Z. Yu, Y. Shi, C. Mu, M. Xu, Deep-irtarget: An automatic target detector in infrared imagery using dual-domain feature extraction and allocation, IEEE Transactions on Multimedia.
[23] S. Yang, L. Liu, M. Xu, Free lunch for few-shot learning: Distribution calibration, in: International Conference on Learning Representations, 2021. url:https://openreview.net/forum?id=JWOiYxMG92s.
[24] Y. Chen, X. Wang, Z. Liu, H. Xu, T. Darrell, A new meta-baseline for few-shot learning, arXiv preprint arXiv:2003.04390.
[25] M. Goldblum, S. Reich, L. Fowl, R. Ni, V. Cherepanova, T. Goldstein, Unraveling meta-learning: Understanding feature representations for few-shot tasks, in: ICML, 2020.
[26] S. Ravi, H. Larochelle, Optimization as a model for few-shot learning, ICLR, in, 2017.
[27] C. Finn, P. Abbeel, S. Levine, Model-agnostic meta-learning for fast adaptation of deep networks, in: ICML, 2017.
[28] Z. Li, F. Zhou, F. Chen, H. Li, Meta-sgd: Learning to learn quickly for few shot learning, CoRR arXiv:1707.09835.
[29] Y. Wang, R.B. Girshick, M. Hebert, B. Hariharan, Low-shot learning from imaginary data, in: CVPR, 2018.
[30] S. Qiao, C. Liu, W. Shen, A.L. Yuille, Few-shot image recognition by predicting parameters from activations, in: CVPR, 2018.
[31] A.A. Rusu, N.C. Rabinowitz, G. Desjardins, H. Soyer, J. Kirkpatrick, K. Kavukcuoglu, R. Pascanu, R. Hadsell, Progressive neural networks, CoRR abs/1606.04671.
[32] O. Vinyals, C. Blundell, T. Lillicrap, K. Kavukcuoglu, D. Wierstra, Matching networks for one shot learning, in: NeurIPS, 2016.
[33] J. Snell, K. Swersky, R.S. Zemel, Prototypical networks for few-shot learning, in: NeurIPS, 2017.
[34] P.W. Battaglia, J.B. Hamrick, V. Bapst, A. Sanchez-Gonzalez, V.F. Zambaldi, M. Malinowski, A. Tacchetti, et al., Relational inductive biases, deep learning, and graph networks, CoRR abs/1806.01261. arXiv:1806.01261.
[35] L. Sang, M. Xu, S. Qian, M. Martin, P. Li, X. Wu, Context-dependent propagating based video recommendation in multimodal heterogeneous information networks, IEEE Transactions on Multimedia.
[36] Z. Wang, M. Xu, N. Ye, F. Xiao, W. none Ruchuan, H. Huang, Computer vision-assisted 3d object localization via cots rfid devices and a monocular camera, IEEE Transactions on Mobile Computing.
[37] R. Zhang, L. Wu, Y. Yang, W. Wu, Y. Chen, M. Xu, Multi-camera multi-player tracking with deep player identification in sports video, Pattern Recognition 102 (2020) 107260.
[38] G.-S. Xie, L. Liu, F. Zhu, F. Zhao, Z. Zhang, Y. Yao, J. Qin, L. Shao, Region graph embedding network for zero-shot learning, in: European Conference on Computer Vision, Springer, 2020, pp. 562–580.
[39] X. Lu, W. Wang, J. Shen, D. Crandall, L. Van Gool, Segmenting objects from relational visual data, IEEE Transactions on Pattern Analysis and Machine Intelligence.
[40] G.-S. Xie, J. Liu, H. Xiong, L. Shao, Scale-aware graph neural network for few-shot semantic segmentation, in: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, 2021, pp. 5475–5484.
[41] N. Courty, R. Flamary, D. Tuia, A. Rakotomamonjy, Optimal transport for domain adaptation, CoRR abs/1507.00504.
[42] Z. Su, Y. Wang, R. Shi, W. Zeng, J. Sun, F. Luo, X. Gu, Optimal mass transport for shape matching and comparison, IEEE Trans. Pattern Anal. Mach. Intell. 37 (11) (2015) 2246–2259.
[43] R. Zhang, C. Mu, Y. Yang, L. Xu, Research on simulated infrared image utility evaluation using deep representation, Journal of Electronic Imaging 27 (1) (2018) 013012.
[44] H. Xu, D. Luo, L. Carin, Scalable gromov-wasserstein learning for graph partitioning and matching, in: Advances in Neural Information Processing Systems 32, Curran Associates Inc, 2019, pp. 3052–3062.
[45] H. Xu, D. Luo, H. Zha, L.C. Duke, Gromov-Wasserstein learning for graph matching and node embedding, in: ICML, 2019.
[46] W. Wang, J. Shen, F. Porikli, Saliency-aware geodesic video object segmentation, in: Proceedings of the IEEE conference on computer vision and pattern recognition, 2015, pp. 3395–3402.
[47] Q. Zhang, L. Ge, R. Zhang, G.I. Metternicht, Z. Du, J. Kuang, M. Xu, Deep-learning-based burned area mapping using the synergy of sentinel-1&2 data, Remote Sensing of Environment 264 (2021) 112575.
[48] M. Zhang, W. Li, R. Tao, H. Li, Q. Du, Information fusion for classification of hyperspectral and lidar data using ip-cnn, IEEE Transactions on Geoscience and Remote Sensing.
[49] C. Dong, Y. Shen, Y. Qu, K. Wang, J. Zheng, Q. Wu, F. Wu, Uavs as an intelligent service: Boosting edge intelligence for air-ground integrated networks, IEEE Network 35 (4) (2021) 167–175.
[50] M. Cuturi, Sinkhorn distances: Lightspeed computation of optimal transport, in: C.J.C. Burges, L. Bottou, M. Welling, Z. Ghahramani, K.Q. Weinberger (Eds.), Advances in Neural Information Processing Systems 26, 2013, pp. 2292–2300.
[51] K. He, X. Zhang, S. Ren, J. Sun, Deep residual learning for image recognition, in: Proceedings of the IEEE conference on computer vision and pattern recognition, 2016, pp. 770–778.
[52] S. Zagoruyko, N. Komodakis, Wide residual networks, arXiv preprint arXiv:1605.07146.
[53] H. Li, D. Eigen, S. Dodge, M. Zeiler, X. Wang, Finding task-relevant features for few-shot learning by category traversal, in: CVPR, 2019.
[54] O. Russakovsky, J. Deng, H. Su, J. Krause, S. Satheesh, S. Ma, Z. Huang, A. Karpathy, A. Khosla, M. Bernstein, et al., Imagenet large scale visual recognition challenge, International journal of computer vision 115 (3) (2015) 211–252.
[55] M. Zhang, W. Li, Q. Du, L. Gao, B. Zhang, Feature extraction for classification of hyperspectral and lidar data using patch-to-patch cnn, IEEE Transactions on Cybernetics 50 (1) (2018) 100–111.

[56] S. Li, W. Deng, Reliable crowdsourcing and deep locality-preserving learning for unconstrained facial expression recognition, IEEE Transactions on Image Processing 28 (1) (2018) 356–370.
[57] H. Pei, B. Yang, J. Liu, K. Chang, Active surveillance via group sparse bayesian learning, IEEE Transactions on Pattern Analysis and Machine Intelligence doi:10.1109/TPAMI.2020.3023092.
[58] W.-Y. Chen, Y.-C. Liu, Z. Kira, Y.-C. Wang, J.-B. Huang, A closer look at few-shot classification, in: ICLR, 2019.
[59] A.A. Rusu, D. Rao, J. Sygnowski, O. Vinyals, R. Pascanu, S. Osindero, R. Hadsell, Meta-learning with latent embedding optimization, in: ICLR, 2019.
[60] Y. Wang, C. Xu, C. Liu, L. Zhang, Y. Fu, Instance credibility inference for few-shot learning, in: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, 2020.
[61] R. Dian, S. Li, X. Kang, Regularizing hyperspectral and multispectral image fusion by cnn denoiser, IEEE Transactions on Neural Networks and Learning Systems 32 (3) (2020) 1124–1135.

Ruiheng Zhang received the B.E. degree in 2014 from Beijing Institute of Technology, China. He was a dual-Ph. D. from University of Technology Sydney, Australia and Beijing Institute of Technology, China. He is an Assistant Professor in Beijing Institute of Technology. He is the author of more than 20 research papers and one book. His current research interests include deep learning, object detection, and remote sensing.

Shuo Yang received the B.E. degree in computer science and technology from the Harbin Institute of Technology, Harbin, China, in 2020. He is currently pursuing a Ph.D. degree in the School of Electrical and Data Engineering, Faculty of Engineering and Information Technology, University of Technology Sydney, advised by Prof.Min Xu. His research lies in computer vision and machine learning.

Qi Zhang received the B.E. degree in Information Engineering and the M.E. degree in Electronic and Communication Engineering from the Beijing Institute of Technology, Beijing, China, in 2014 and 2016. She is currently pursuing the Ph.D. degree in Remote Sensing at the University of New South Wales (UNSW), Sydney, Australia. Her research interests include the deep-learning-based algorithm design and its applications on the earth observation of Synthetic Aperture Radar and multi-spectral remote sensing.

Lixin Xu received his PhD degree in information engineering from Harbin Institute of Technology. He is a Professor in Beijing Institute of Technology. He has published 100 journal and conference papers. His current research interests include deep learning, MEMS, and target detection. He has served as the Editor and Reviewer for several international journals and conferences. He is the editorial board of Journal of Detection and Control.

Yang He received the BS degree and MSc from University of Science and Technology of China, Hefei, China, in 2014 and 2017, respectively. He is currently working toward the PhD degree with the Australian Artificial Intelligence Institute (AAII), Faculty of Engineering and Information Technology, University of Technology Sydney. His research interests include deep learning, computer vision, and machine learning.

Fan Zhang received the B.E. degree in communication engineering from the Civil Aviation University of China, Tianjin, China, in 2002, the M.S. degree in signal and information processing from Beihang University, Beijing, China, in 2005, and the Ph.D. degree in signal and information processing from the Institute of Electronics, Chinese Academy of Sciences, Beijing, in 2008. He is a Full Professor in electronic and information engineering with the Beijing University of Chemical Technology, Beijing. His research interests include remote sensing image processing, high-performance computing, and artificial intelligence. Dr. Zhang is an Associate Editor for IEEE ACCESS and a Reviewer for the IEEE TRANSACTIONS ON GEOSCIENCE AND REMOTE SENSING, the IEEE JOURNAL OF SELECTED TOPICS IN APPLIED EARTH OBSERVATIONS AND REMOTE SENSING, the IEEE GEOSCIENCE AND REMOTE SENSING LETTERS, and the Journal of Radars.