Transductive Relation-Propagation With Decoupling Training for Few-Shot Learning

Yuqing Ma, Shihao Bai, Wei Liu, Shuo Wang, Yue Yu, Xiao Bai, Xianglong Liu, *Member, IEEE*, and Meng Wang, *Fellow, IEEE*

***Abstract*— Few-shot learning, aiming to learn novel concepts from one or a few labeled examples, is an interesting and very challenging problem with many practical advantages. Existing few-shot methods usually utilize data of the same classes to train the feature embedding module and in a row, which is unable to learn adapting to new tasks. Besides, traditional few-shot models fail to take advantage of the valuable relations of the support-query pairs, leading to performance degradation. In this article, we propose a transductive relation-propagation graph neural network (GNN) with a decoupling training strategy (TRPN-D) to explicitly model and propagate such relations across support-query pairs, and empower the few-shot module the ability of transferring past knowledge to new tasks via the decoupling training. Our few-shot module, namely TRPN, treats the relation of each support-query pair as a graph node, named relational node, and resorts to the known relations between support samples, including both intraclass commonality and interclass uniqueness. Through relation propagation, the model could generate the discriminative relation embeddings for support-query pairs. To the best of our knowledge, this is the first work that decouples the training of the embedding network and the few-shot graph module with different tasks, which might offer a new way to solve the few-shot learning problem. Extensive experiments conducted on several benchmark datasets demonstrate that our method can significantly outperform a variety of state-of-the-art few-shot learning methods.**

***Index Terms*— Decoupling training, few-shot learning, knowledge transfer, relation-propagation graph.**

I. Introduction

RECENTLY, deep learning [1]–[4] has greatly promoted the development of computer vision. However, traditional deep learning models [5]–[8] gain knowledge from millions of data, making it hard to rapidly transfer to novel concepts as human intelligence. Thus, few-shot learning [9]–[15] has been designed for this purpose, aiming at learning new concepts from only one or a few examples. It is a meaningful but quite challenging problem with many practical advantages, *e.g.*, developing real-time interactive vision applications for portable devices and transferring knowledge from existing models to novel categories without retraining.

There have been a few attempts in few-shot learning area in the past years, trying to extract the information from the limited labeled data and make accurate predictions on the unknown ones. One promising learning paradigm is a metric-based solution, which exploits the feature similarity information by embedding both support and query samples into a shared feature space. For instance, Matching Net [16] combined attention and memory together to map an unlabeled query sample to its label relying on few labeled support samples, and introduced the episodic training mechanism into few-shot learning. In [17], a Prototypical Network learned a discriminative metric space where classification was performed by calculating the distance between the query sample and the prototype representation of each class, which helped reduce the intraclass variations. In [18], a Covariance Metric Network constructed an embedded local covariance representation to extract the second-order statistic information of each class and defined a new deep covariance metric to measure the consistency of distributions between query samples and support samples for the few-shot classification tasks.

Actually, to meet the requirement of learning novel concepts from a few labeled samples, one should focus on revealing the accurate relations between support-query samples in a few-shot task, especially in metric-based methods. Recently, graph-based approaches [19]–[21] have been proposed, promoting the few-shot learning field. Those methods treat each sample as a graph node and represent the support-query relations through edges. Although they have shown promising performance to address the few-shot learning task, without directly modeling the relations of the support-query pairs, the underlying information shared across different support-query pairs suffers from severe underutilization, inevitably leading to inferior performance.

Besides, previous methods [22]–[24] usually pretrained the low-level feature embedding network on the large-scale base classes as a regular multiclass classifier, following which the specifically designed few-shot module would be introduced to further enhance the learning ability from a few samples using

Manuscript received 25 November 2020; revised 14 March 2021; accepted 19 May 2021. Date of publication 17 June 2021; date of current version 28 October 2022. This work was supported in part by the National Key Research and Development Plan of China under Grant 2020AAA0103502, in part by the National Natural Science Foundation of China under Grant 62022009 and Grant 61872021, in part by the Beijing Nova Program of Science and Technology under Grant Z191100001119050, and in part by the Academic Excellence Foundation of the Beijing University of Aeronautics and Astronautics (BUAA) for Ph.D. Students. *(Corresponding author: Xianglong Liu.)*

Yuqing Ma, Shihao Bai, Wei Liu, Shuo Wang, and Xiao Bai are with the State Key Laboratory of Software Development Environment, Beihang University, Beijing 100191, China.

Yue Yu is with the Key Laboratory of Parallel and Distributed Computing, College of Computer, National University of Defense Technology, Changsha 410073, China.

Xianglong Liu is with the State Key Laboratory of Software Development Environment, Beihang University, Beijing 100191, China, and also with Beijing Advanced Innovation Center for Big Data-Based Precision Medicine, Beihang University, Beijing 100191, China (e-mail: xlliu@nlsde.buaa.edu.cn).

Meng Wang is with the School of Computer and Information Science, Hefei University of Technology, Hefei 230009, China.

Color versions of one or more figures in this article are available at https://doi.org/10.1109/TNNLS.2021.3082928.

Digital Object Identifier 10.1109/TNNLS.2021.3082928

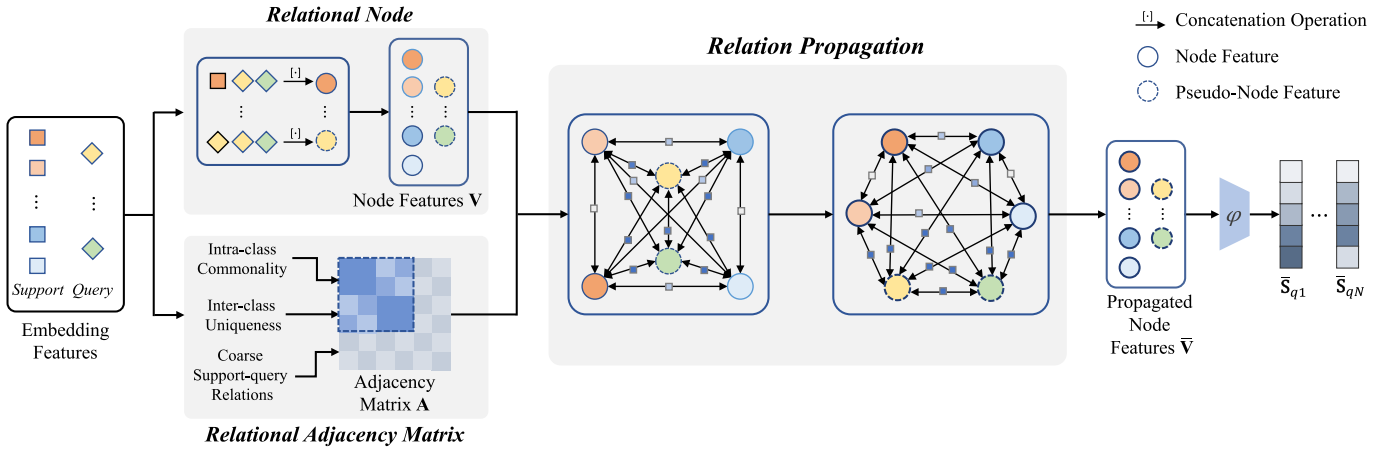

Fig. 1. Illustration of the few-shot module TRPN. We first initialize the relational nodes by the feature concatenation of each support sample (or query sample for the pseudo relational node) and the entire query set and construct the relational adjacency matrix according to the known relations of support set, including both intraclass commonality and interclass uniqueness. The coarse relations between support-query samples are also considered in the adjacency matrix. Subsequently, the relation propagation is performed to pursue the discriminative relational node features, which are utilized to compute the final classification scores through a similarity function φ.

the training data of the same categories. It has been shown that pretraining on large-scale base classes could accelerate and stabilize the training process and bring obvious gains to few-shot classification performance. However, during the training phase, the tasks accomplished by the few-shot module are subtasks that have been already seen by the multiclassifier. That is to say, the traditional training strategy cannot empower few-shot models with the ability of transferring knowledge to new tasks.

To address these problems, in this article, we propose a transductive relation-propagation graph neural network (GNN) with decoupling training strategy (TRPN-D). The proposed framework captures the relations in the task through the few-shot module dubbed TRPN, and meanwhile explicitly learns to transfer knowledge from seen tasks to novel tasks with the decoupling training strategy.

Different from the previous graph model where each graph node represents a sample and support-query relations are computed after propagation, TRPN directly models and propagates the sample relations across the support-query pairs through the relational graph, as shown in Fig. 1. Specifically, each graph node represents the sample relation of the corresponding support-query pair, dubbed relational node, and thus the few-shot learning is converted into a simple node classification task. On the premise that similar support samples from the same class often share similar relations with the same query, our TRPN resorts to the known relations between support samples to learn more discriminative relational representations for support-query pairs. Both intraclass commonality and interclass uniqueness of the support set are propagated and aggregated on the relational graph, in order to better measure the relational adjacency among the different support-query pairs. With the relational graph, both sample similarities and the support-query relations are exploited, obviously improving the learning performance.

Furthermore, to empower the few-shot module the ability to transfer knowledge to new tasks and simulate the few-shot setting as much as possible, a decoupling training strategy is introduced to decouple the training of the embedding network and the few-shot module with different tasks. The proposed strategy ensures the TRPN better extracts the relations on new tasks unseen to the feature embedding network, and in the meanwhile maintains the discriminative representation ability of the feature embedding module. Specifically, we divide the whole training dataset into different groups on which we construct different tasks, respectively, train the feature embedding modules and the proposed TRPN [25]. The data utilized to train few-shot metric-learning module contains certain categories which are unseen to the feature embedding module, therefore forming new tasks which are previously unseen. The abovementioned strategy adopts diversified classes to train the feature embedding module to capture rich semantic feature representation, while further help TRPN to capture and propagate unseen relations, bridging the semantic gap between large-scale base samples and a few novel samples.

To the best of our knowledge, this is the first work that explicitly decouples the training of the embedding network and the few-shot module with different tasks, which might offer a new way to solve the few-shot learning problem. Note that we extend our prior conference publication [25] which models and propagates sample relations across support-query pairs in a transductive way with traditional training strategy (TRPN-T for the shot). In this article, with the newly proposed decoupling training strategy, our TRPN-D could learn to transfer knowledge from past experience to new observations, which is more suitable for a few-shot learning scenario. Experiments show that TRPN-D could beat the TRPN-T by a significant margin in most cases, validating our claim that decoupling training strategy could help bridge the semantic gap and empower the few-shot module the ability to transfer knowledge to new tasks. Moreover, we discuss different implementations for the decoupling training strategy, ensuring our implementation is theoretically reasonable and yields competitive performance.

Extensive experiments were conducted on several benchmark datasets such as *mini*ImageNet [16] and *tiered*ImageNet [26], and the results demonstrate that our method can drastically boost the classification accuracy and achieve the new state-of-the-art on both 5-way-1-shot and 5-way-5-shot tasks. We confirm the effectiveness and robustness of the proposed model with different backbone architectures which witness significant performance gains compared to the other few-shot approaches.

II. Related Work

To tackle the few-shot learning task, a variety of methods have been proposed, which can be roughly divided into three types, i.e., optimization-based, generation-based, and metric-based methods.

The optimization-based solution, as one of the most popular few-shot learning paradigms, tries to capture the relation information among the tasks, leveraging the previous learning experience as a prior over tasks. For instance, Santoro *et al.* [27] trained an long-short term memory (LSTM) as a controller to interact with an external memory module. And the work [28] adopted an LSTM-based meta-learner as an optimizer which is trained to converge a learner classifier to a good solution quickly on each task. Meanwhile, Mishra *et al.* [29] combined temporal convolutions with soft attention, which enables the meta-learner to aggregate contextual information from the past experience and pinpoint specific pieces of information within that context. In [30], the authors introduced a first-order gradient-based meta-learning algorithm named Reptile, whose training process is similar to joint training. Recently, Flennerhag *et al.* [31] proposed a meta-learner named Warp-Grad, which meta-learns to precondition gradients defined point-wise in parameter space and facilitates gradient descent during task adaptation. And Elsken *et al.* [32] integrated gradient-based meta-learning with neural architecture search, and proposed a meta-learning algorithm that could quickly adapt the meta-architecture to a task-dependent architecture based on few labeled data points with only a few steps of gradient descent. However, as pointed out by Rusu *et al.* [24], while these approaches iterate over samples from all classes in their updates, they lack the capability of learning effective embeddings.

Generation-based methods adopt a meta-learner for few-shot data augmentation or learn to predict classification weights for novel classes. In [33], the few-shot training set is expanded via a "hallucinator," whose output is then provided to the meta-learner for model optimization. For weights generation, Gidaris and Komodakis [34] proposed a few-shot object recognition system capable of dynamically learning novel categories from only a few training data while does not forget the base categories, leading to feature representations that generalize better on unseen categories. Qiao *et al.* [35] proposed to learn a category-agnostic mapping from activations to parameters so that the parameters for novel categories can be predicted by a simple forward pass. Recently, Gidaris and Komodakis [36] employed a Denoising Autoencoder network which takes a set of classification weights corrupted with Gaussian noise as input and learns to reconstruct the target-discriminative classification weights. In [37], the proposed method learned to generate optimal classification weights by maximizing the mutual information between generated weights and support/query data. However, most of these methods do not consider generating different classification weights by dividing the training set data into different groups or taking advantage of the information between different query samples.

A metric-based solution serves as another promising few-shot learning paradigm, which exploits the feature similarity information by embedding both support and query samples into a shared feature space. Some earlier studies address the few-shot learning problem by learning a proper distance or similarity metric, such as cosine similarity [16], Euclidean distance [17], and deep nonlinear relation network [38]. In [39], a Deep Nearest Neighbor Neural Network is proposed to replace the image-level feature-based measure in the final layer with a local descriptor-based image-to-class measure. And Li *et al.* [18] designed a local covariance representation which is used to measure the distribution consistency for the few-shot classification tasks. Recently, Li *et al.* [40] proposed an adaptive margin principle to enhance the discriminative power of embedding features, especially for similar classes. And Simon *et al.* [41] extended existing dynamic classifiers by using subspaces and introduced a discriminative formulation to maximize the discrimination between subspaces. Moreover, due to the great potential of graph networks in relation to modeling and propagation, a few approaches adopted graph networks to handle the few-shot learning problem. Specifically, Satorras and Estrach [19] proposed to first construct a graph where each example of the support set and query set is considered as a graph node. Liu *et al.* [20] proposed a transductive propagation network on the node features obtained from a deep neural network. Kim *et al.* [21] proposed an edge-labeling GNN learning to predict the edge-labels rather than the node-labels on the graph. Instead of modeling instance-level relations of examples, Yang *et al.* [42] constructed a dual complete graph network composed of a point graph and a distribution graph, which combines the distribution-level relations and instance-level relations. Nevertheless, all these graph-based methods do not model the relations of the support-query pairs directly, which inevitably leads to the severe underutilization of the underlying information shared across different support-query pairs.

III. Approach

Before introducing our transductive relation-propagation GNN with decoupling training (TRPN-D), we briefly review the preliminary of few-shot setting in III-A. As the transductive relation-propagation GNN (TRPN) lies at the core of the proposed method, we next elaborate the details of the relational graph including both the nontransductive setting and the transductive setting in Sections III-B and III-C, respectively. After that, we illustrate the decoupling training strategy in Section III-D. Fig. 2 is a schematic depiction of the proposed learning pipeline. At the end of this section, we present the learning and inference process of the proposed model.

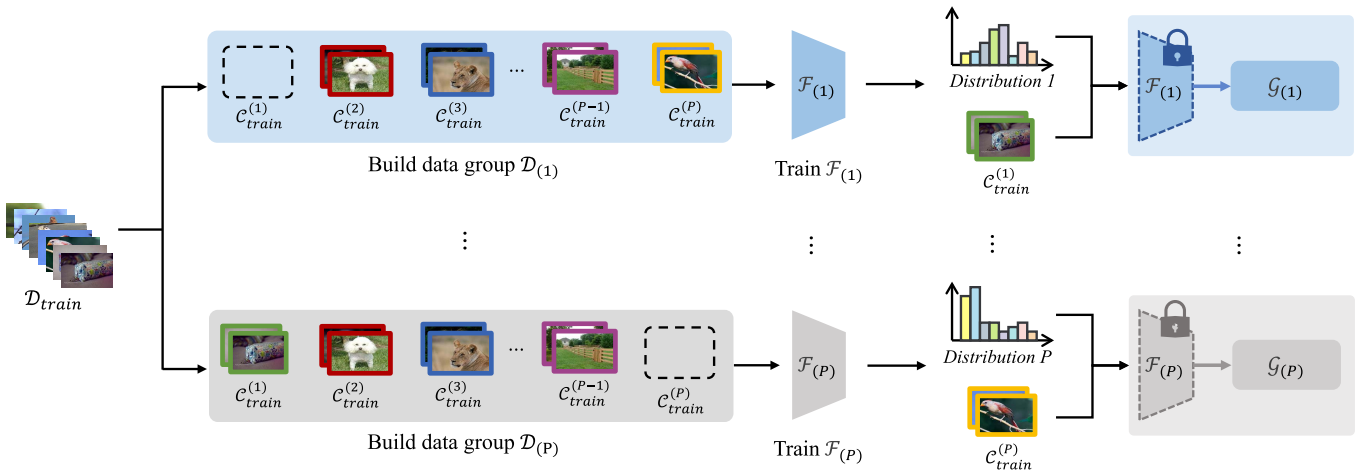

Fig. 2. Schematic depiction of the proposed TRPN-D. The whole training classes are divided into P disjoint partitions. And the whole training dataset $\mathcal{D}_{\text{train}}$ is divided into P groups, with the pth group removing samples of $\mathcal{C}_{(p)}$. The pth embedding module captures the latent embedding of data distribution $\mathcal{D}_{(p)}$, after which the samples of the whole dataset, containing samples of $\mathcal{C}_{(p)}$, are fed into the proposed model to train the few-shot metric-learning module $\mathcal{G}_{(p)}$.

A. Preliminary

Mathematically speaking, few-shot learning is to determine the class label of each unlabeled sample in the query set $\mathcal{Q}$ based on the support set $\mathcal{S}$ unseen before, where $\mathcal{S}$ contains N different image classes $(\mathcal{C}_1, \ldots, \mathcal{C}_N)$ and K (K is small, *e.g.*, $K = 5$) labeled samples per class. This setting is also called N-way K-shot classification.

Previous few-shot methods usually follow the episodic training which constructs N-way K-shot tasks similar to that of test phase using a relatively large labeled dataset $\mathcal{D}_{\text{train}} = \{\mathbf{x}, \mathbf{y}\}$ with a set of class labels $\mathcal{C}_{\text{train}}$, where $\mathbf{x}$ is the sample and $\mathbf{y} \in \mathcal{C}_{\text{train}}$ is the corresponding label. A low-level feature embedding network $\mathcal{F}$ is first trained as a multiclass classifier using cross entropy to generate discriminative and informative feature embeddings. After that, the output of $\mathcal{F}$ will be fed into the few-shot module $\mathcal{G}$ for further adaptation to the current few-shot task.

To construct the N-way K-shot task Γ, in each episode, a small subset of N classes are sampled from $\mathcal{C}_{\text{train}}$, where $\mathcal{S} = \{(\mathbf{x}_i, y_i)\}_{i=1}^{T_\mathcal{S}}$ and $\mathcal{Q} = \{(\mathbf{x}_i, y_i)\}_{i=T_\mathcal{S}+1}^{T_\mathcal{S}+T_\mathcal{Q}}$. The subscript i denotes the ith sample. Also we define $T_\mathcal{S}$, $T_\mathcal{Q}$, $|\Gamma| = T_\mathcal{S} + T_\mathcal{Q}$ as the total number of the samples in the support set $\mathcal{S}$, query set $\mathcal{Q}$, and task Γ. The objective is to train a classifier to make predictions on $\mathcal{Q}$ through learning the limited labeled support set $\mathcal{S}$. Thus, when transferring to an unseen set of novel classes $\mathcal{C}_{\text{test}}$, the model can also work well.

B. Relation-Propagation Graph Network

In this section, we illustrate the details of our relation-propagation module which explicitly models the relations between support-query pairs. Previous GNN-based few-shot learning models utilize each node to represent a data sample, and simply pass messages between samples, ignoring the valuable information shared among different support-query pairs. The underutilization of information inevitably leads to inferior performance. Therefore, in the proposed TRPN, each node represents the relation of a support-query pair, and such relations of support-query pairs are propagated to further refine the relational embeddings. In this way, the few-shot learning can be cast into a node classification problem. The adequate exploration of the task helps the few-shot module make accurate predictions on the query samples.

Specifically, for each sample $\mathbf{x}_i$ of the task Γ, a convolutional embedding network is first employed to extract the feature representation $\mathbf{g}_i$. Subsequently, a fully connected graph $\mathcal{G} = (\mathcal{V}, \mathcal{A}; \Gamma)$ is initially constructed to characterize the relations of the task Γ, where $\mathcal{V}$ and $\mathcal{A}$ denote the set of relational nodes and relational adjacency matrix of the graph, respectively. It is worth noting that the model processed each query of $\mathcal{Q} = \{(\mathbf{x}_i, y_i)\}_{i=T_\mathcal{S}+1}^{T_\mathcal{S}+T_\mathcal{Q}}$ independently, without considering the relationship between the query samples, under nontransductive setting. Therefore, we took a query sample $\mathbf{x}_q$ as an example to introduce our relational graph model. The computation will be performed for each query to get the final result.

1) Relational Node: To explicitly model the sample relations between support-query pairs, we introduce the relational nodes indicating the relation embeddings of support-query pairs. The relational node embedding matrix is termed as $\mathbf{V}$, where $\mathbf{V}_i$ indicates the embedding of the ith node. Given a query sample $\mathbf{x}_q$ where $q \in \{T_\mathcal{S}+1, \ldots, |\Gamma|\}$, $\mathbf{V}_i$ is initialized by the concatenation of the support feature $\mathbf{g}_i$ and query feature $\mathbf{g}_q$

$$\mathbf{V}_i = [\mathbf{g}_i, \mathbf{g}_q] \tag{1}$$

where $i = \{1, \ldots, T_\mathcal{S}\}$, and $[\cdot, \cdot]$ denotes the concatenation operation. Specially, we name the first concatenated support sample of a relational node as the dominant sample based on which to determine the relational adjacency.

Though the support sample dominates the learning of relations of support-query pairs, without considering the characteristic of the query sample, the relational node embeddings

will discard the raw, yet important relational information contained between support and query samples, and thus causes information underutilization. To avoid this issue, we further introduce pseudo relational nodes so as to actively involve the query samples in the graph. A pseudo relational node is defined as

$$\mathbf{V}_q = \left[\mathbf{g}_q, \mathbf{g}_q\right]. \tag{2}$$

Without loss of generality, the first concatenated sample is also named as the dominant sample of the pseudo relational node.

2) Relational Adjacency Matrix: The relational adjacency matrix reflects the adjacency between two relational nodes and measures the relation-propagation degree (namely how much information should be propagated). Since similar support samples usually share similar relations with the same query sample, we can naturally estimate node adjacency according to the known relations (through labels) of the support set. We denote the relational adjacency $\mathbf{A}_{ij}$ as the element of adjacency matrix $\mathbf{A}$ on the graph, representing the coarse relations between dominant samples i and j of relational nodes $\mathbf{V}_i$ and $\mathbf{V}_j$. The adjacency matrix should manifest the known information (label) of the support set, including both intraclass commonality and interclass uniqueness, and thus weighted propagate the relations to guide the learning of discriminative relation embeddings, with the consideration of query characteristics.

Hence, we define the adjacency between relational nodes according to the relation of their dominant samples as

$$\mathbf{A}_{ij} = \begin{cases} 1, & \text{if } \ i, j \leqslant T_{\mathcal{S}}, \ l_{ij} = 1 \\ -\phi([\mathbf{g}_i, \mathbf{g}_j]), & \text{if } \ i, j \leqslant T_{\mathcal{S}}, \ l_{ij} = 0 \\ \phi([\mathbf{g}_i, \mathbf{g}_j]), & \text{if } \ i = q \text{ or } j = q \end{cases} \tag{3}$$

where l_{ij} is the associate-label for each node, defined according to ground truth labels of consisting samples

$$l_{ij} = \begin{cases} 1, & \text{if } y_i = y_j \\ 0, & \text{otherwise} \end{cases} \tag{4}$$

and $\phi(\cdot)$ is the adjacency function. There are a number of candidate implementations for adjacency function, such as cosine similarity, **negative** squared Euclidean distance, or neural network, *etc*. The adjacency function $\phi(\cdot)$ simply indicates the adjacency of two inputs and further represents how much information would be propagated. In contrast, inverting the adjacency scores between two samples could help diminish the commonality between two samples and learn the uniqueness of each sample through propagation.

From Formula 3, we can see three kinds of relations: a) for two relational nodes whose dominant support samples come from the same class, we preserve their intraclass commonality through simply using the corresponding associate-label as the adjacency score. b) for two nodes whose dominant support samples come from different classes, we use the adjacency function to evaluate the adjacency between the two samples and invert the result. Therefore, the more similar two different classes are the more commonality of the relation each class will subtract during the feature aggregation. By diminishing the commonality according to the negative adjacency score, it maximizes the uniqueness of the two classes and helps the network to learn the discriminative representations which assemble similar samples and disperse the different ones. c) for two nodes whose dominant samples are support sample and query sample since we do not know the exact relations between the samples as the labeled support samples, we simply computed the coarse predicted relations between support-query pairs through the adjacency function, with the query characteristics involved during relation propagation.

Thus, the known information including both intraclass commonality and interclass uniqueness of support set is considered, and the support-query relations are exploited as well in the construction of the relational adjacency matrix.

3) Relation Propagation: Once we have the weighted graph, we can update the node features to pursue the discriminative relational embeddings for support-query pairs, using the following propagation rule:

$$\overline{\mathbf{V}} = \sigma(\tilde{\mathbf{D}}^{-1/2}\tilde{\mathbf{A}}\tilde{\mathbf{D}}^{-1/2}\mathbf{V}\mathbf{W}). \tag{5}$$

Here, $\tilde{\mathbf{A}} = \mathbf{A} + \mathbf{I}$ is the relational adjacency matrix of the graph added with self-connections. $\mathbf{I}$ is the identity matrix, $\tilde{\mathbf{D}}_{ii} = \sum_j \tilde{\mathbf{A}}_{ij}$ and $\mathbf{W}$ is a trainable weight matrix. σ denotes the activation function. After the relation propagation, the updated relational embedding $\overline{\mathbf{V}}$ becomes more discriminative and could better reflect the relations between support-query pair. Our relational graph only contains one layer. We also tried to add more layers in our experiments, but it brought an obvious performance drop as the number of layers increases. It is much likely that adding more layers would cause an over-smoothing problem [43] where indistinguishable feature representations are generated during propagation and further hurts the classification performance.

After relation propagation and feature aggregation, the similarity level between support-query pair could be measured through a similarity function φ operating on the $\overline{\mathbf{V}}$, and quantized as a similarity score vector. Specifically, for the qth query sample, the similarity score vector $\mathbf{s}_q = \varphi(\overline{\mathbf{V}}_q)$. We conduct the above operation for each query sample and construct the similarity matrix $\mathbf{S}$ composed of their similarity vector, where $\mathbf{S}$ is a $T_{\mathcal{Q}} \times |\Gamma|$ matrix, with $\mathbf{S}_{qi}$ indicating the similarity score of the ith support samples and the query sample $\mathbf{x}_q$. Here, the similarity score can be considered as a probability that the two samples are from the same class. Therefore, each query sample can be classified by simple weighted voting.

C. Transductive Learning

For a novel classification task, while the proposed method could take advantage of the valuable information shared between different support-query pairs and generate discriminative relation embeddings, there are still some fundamental difficulties in learning with scarce data. For the sake of making full use of limited data to further improve the classification performance, some previous work introduced transduction, also known as transductive inference [20], [21], which considers the relations between samples in the query set, so as to

predict them as a whole. Referring to previous graph models, the relations between query samples can be easily propagated on the graph through feature aggregation by involving more relational nodes composed of support-query pairs and pseudo relational nodes.

However, since the relations between dominant samples are the same and indistinguishable regardless of the concatenated query sample, we consider the implementation of transductive learning mentioned above to be computation exhaustive, which will result in performance degradation as well. Hence, in order to simultaneously process the relations between all the support-query pairs within the relational node and during the propagation on the graph, for each relational node, we combine a support sample with the entire query set.

Specifically, under the transductive setting, we initialize the relational node features as

$$\mathbf{V}_i = [\mathbf{g}_i, \mathbf{g}_{T_\mathcal{S}+1}, \mathbf{g}_{T_\mathcal{S}+2}, \ldots, \mathbf{g}_{|\Gamma|}]. \tag{6}$$

Analogously, the pseudo node could be initialized as

$$\mathbf{V}_q = [\mathbf{g}_q, \mathbf{g}_{T_\mathcal{S}+1}, \mathbf{g}_{T_\mathcal{S}+2}, \ldots, \mathbf{g}_{|\Gamma|}] \tag{7}$$

where $i = \{1, \ldots, T_\mathcal{S}\}$, and $q = \{T_\mathcal{S}+1, \ldots, |\Gamma|\}$. The construction of the relational adjacency matrix $\mathbf{A}$ is similar to the one under the nontransductive setting, which indicates pairwise adjacency

$$\mathbf{A}_{ij} = \begin{cases} 1, & \text{if } i, j \leqslant T_\mathcal{S}, l_{ij} = 1 \\ -\phi([\mathbf{g}_i, \mathbf{g}_j]), & \text{if } i, j \leqslant T_\mathcal{S}, l_{ij} = 0 \\ \phi([\mathbf{g}_i, \mathbf{g}_j]), & \text{if } T_\mathcal{S}+1 \leqslant i \leqslant |\Gamma| \\ & \text{or } T_\mathcal{S}+1 \leqslant j \leqslant |\Gamma|. \end{cases} \tag{8}$$

As a consequence, we can follow the same process of relation-propagation as under the nontransductive setting.

Similarly, we employ a similarity function φ to operate on the propagated relational nodes $\overline{\mathbf{V}}$, which outputs a $T_\mathcal{Q} \times |\Gamma|$ similarity matrix $\mathbf{S}$ directly.

D. Decoupling Training Strategy

Traditional training strategy with a pretrained embedding network lays a strong foundation for capturing informative features and is proven helpful for further improving the few-shot classification performance. However, a dilemma is raised to the training phase of few-shot learning task: on the one hand, the few-shot learning module seeks to rapidly capture the similarity relations with limited samples on unseen tasks; on the other hand, during the training phase, the pretrained embedding network $\mathcal{F}$ continually provides features of the seen categories whose similarity relations have been primely captured already, forming seen subtasks and leading to little learning space for the few-shot learning module, not to mention how to handle the new tasks unseen before. Hence, with the traditional training strategy, the few-shot learning module deficiently captures the relations and struggles to transfer knowledge.

To bridge the semantic gap between the seen and unseen tasks, we propose a novel decoupling training strategy for a few-shot learning task, maintaining the discriminative representation ability of the feature embedding module and ensuring the TRPN to better distill unseen relations at the same time. The training set is divided into different groups, guaranteeing that certain classes adopted to train the few-shot learning module are unseen to the low-level feature embedding module even in the training phase, imitating the test scenario.

We initially divide the training classes $\mathcal{C}_{\text{train}}$ into P equal-size disjoint partitions, respectively, termed as $\mathcal{C}_{\text{train}}^{(1)}, \ldots, \mathcal{C}_{\text{train}}^{(P)}$. A simple idea is using different data partitions to, respectively, train the feature embedding module and the few-shot learning module. However, as previous work [44] presented, training feature embedding module on more (diverse) base classes leads to obvious gains. Hence, we group different partitions into a large set to preserve the diversity, in order to generate informative feature representations. Specifically, the pth group is defined as $\mathcal{D}_{(p)} = \{(\mathbf{x}, \mathbf{y}) | \mathbf{y} \in \bigcup_{j \in \{1,\ldots,P\} \wedge j \neq p} \mathcal{C}_{\text{train}}^{(j)}\}$, where $p = 1, \ldots, P$.

After that, P parallel convolutional feature embedding networks are constructed with the same architecture serving as the foundation of generating diverse comparable feature patterns for few-shot learning, followed by their corresponding few-shot module TRPN. We denote the pth feature embedding network as $\mathcal{F}_{(p)}$, on the top of which is multiclassifier. The network $\mathcal{F}_{(p)}$ with the added classifier is trained on $\mathcal{D}_{(p)}$ with standard entropy loss.

Subsequently, we remove the multiclassifier and feed the whole training dataset containing samples of $\mathcal{C}_{\text{train}}^{(p)}$ unseen to $\mathcal{F}_{(p)}$ into the feature embedding networks, which forms new few-shot tasks in the training phase. The output of $\mathcal{F}_{(p)}$ is then used to train the corresponding TRPN, which is denoted as $\mathcal{G}_{(p)}$ for ease of description. Thus, the feature embeddings of $\mathcal{C}_{\text{train}}^{(p)}$ are not as representative as those of the other classes, and further distillation by the successive TRPNs is required to capture the relation through other samples.

Through the decoupling training strategy, the TRPNs could distill representative information through coarse embeddings from $\mathcal{F}_{(p)}$, and sufficiently explore the relations among samples. What's more, it helps develop the ability of TRPNs to effectively transfer mastered knowledge to novel unseen classes, fulfilling the objective of few-shot learning.

E. Learning and Inference

In this section, we will particularly elaborate on the training procedure and illustrate how to make the inference on novel unseen classes.

1) Training Procedure: For the purpose of generating informative feature embeddings, we first train the feature embedding network with standard cross entropy loss using their relative group of data. The classification loss function can be defined as

$$\ell_\mathcal{F} = \sum_{p=1}^{P} \frac{1}{|\mathcal{D}_{(p)}|} \sum_{(\mathbf{x},\mathbf{y}) \in \mathcal{D}_{(p)}} - \sum_{c=1}^{\frac{P-1}{P}|\mathcal{C}_{\text{train}}|} \mathbf{y}_\mathbf{c} \log(\tilde{\mathbf{y}}_c)$$

where $\tilde{\mathbf{y}} = \mathcal{F}_{(p)}(\mathbf{x})$, $|\mathcal{D}_{(p)}|$ is the number of data samples in the pth group.

Then, we remove the classifier layer of each feature embedding network, and randomly sample a set of class from $\mathcal{C}_{\text{train}}$

to construct N-way K-shot few-shot tasks to further train the TRPNs. As a result, the similarity matrix of TRPN $\mathcal{G}_{(p)}$ is termed as $\mathbf{S}^{(p)}$. To guide the learning of TRPNs, we first introduce the relational node classification loss

$$\ell_r = \sum_{p=1}^{P} \sum_{i=1}^{|\Gamma|} \sum_{q=T_S+1}^{|\Gamma|} l_{qi} \log \mathbf{S}_{qi}^{(p)} + (1 - l_{qi}) \log\left(1 - \mathbf{S}_{qi}^{(p)}\right).$$

The node loss is to promise the node feature could represent the relationship between samples inside the relational node, and guide the model to generate discriminative relation embeddings for further classification.

Besides, if the adjacency function ϕ is implemented by neural networks, we could further take advantage of the label information of support and query samples to enhance the capability of ϕ as follows:

$$\ell_\phi = \sum_{p=1}^{P} \sum_{i=1}^{|\Gamma|} \sum_{j=1}^{|\Gamma|} l_{ij} \log \phi^{(p)}\left(\left[\mathbf{g}_i^{(p)}, \mathbf{g}_j^{(p)}\right]\right) + (1 - l_{ij}) \log\left(1 - \phi^{(p)}\left(\left[\mathbf{g}_i^{(p)}, \mathbf{g}_j^{(p)}\right]\right)\right).$$

As a result, the final loss function to train the TRPNs is

$$\ell_{\mathcal{G}} = \ell_r + \lambda \ell_\phi$$

where λ is the hyperparameter to balance the weights between the two loss functions.

2) Inference on Novel Class: In the test phase, we evaluate the performance of the proposed method on novel categories that are not seen in episodic training. We perform the relation propagation procedure and compute the query-to-class similarity score that a query sample $\mathbf{x}_q$ belongs to the nth category of each TRPN through weighted voting, and then compute the average scores of P parallel TRPNs

$$\overline{\mathbf{S}}_{qn} = \frac{1}{P} \sum_{p=1}^{P} \sum_{\{i|(\mathbf{x}_i, y_i) \in \mathcal{S} \wedge y_i = \mathcal{C}_n\}} \mathbf{S}_{qi}^{(p)}. \tag{9}$$

The category with the highest score is regarded as the final prediction.

IV. Experiments

In this section, we investigate our TRPN-D on two widely used datasets. We first introduce the experimental setup, after which we report the few-shot classification performance to show the effectiveness of the proposed TRPN-D compared with a number of state-of-the-art few-shot approaches. Then we show the effects of different decoupling implementation. At the end of this section, we assess the impact of each component of our TRPN-D.

A. Experimental Setup

1) Datasets: We employ the widely used datasets in prior studies, including *mini*ImageNet dataset [16] and *tiered*ImageNet dataset [26]. Both datasets contain various red green blue (RGB) colorful images of size 84×84. The *mini*ImageNet dataset consists of 100 classes, each of which contains 600 samples. The *tiered*ImageNet has a larger data size compared to the *mini*ImageNet, which contains a total of 608 classes and 779 165 RGB color images. On average, each class contains 1281 images. We follow the commonly used split setting for training, validation, and test sets. The classes contained in three splits are disjoint, that is, all classes in the validation and test set are invisible to the model during training. According to [26], the split in *tiered*ImageNet is more challenging, due to that the test classes are less similar to the training ones, which represents a more realistic few-shot learning scenario.

2) Implementation Details: Standard data augmentation including random crop, left-right flip, and color jitter is applied in the training phase. To further reduce the intraclass variations and obtain more generalized features for novel classes, we adopt a cosine-similarity-based multiclassifier [34], [52] instead of a fully connected layer to classify the base classes during the training procedure of the feature embedding networks. In addition, we follow [53] and introduce self-supervision into the training procedure by adding an auxiliary self-supervised loss. While training the feature embedding networks, the number of parallel feature embedding networks P is set to 4 in our experiments. We further investigate the effect of different values of P on the accuracy in Section IV-C. For the training procedure of the TRPNs, we adopt the episodic training procedure following [16]. The number of training iterations on *mini*ImageNet and *tiered*ImageNet are 100 and 200 K, respectively. We use Adam optimizer [54] with an initial learning rate of 0.001, and reduce the learning rate by half every 15 and 30 K iterations, respectively, on *mini*ImageNet and *tiered*ImageNet. The weight decay is set to $1e^{-6}$. The mini-batch size for all experiments is 10. We use the validation set to select the training episodes with the best accuracy. We take three implementations for adjacency function and similarity function, namely negative squared Euclidean distance, and neural network, which we will further illustrate in Section IV-B. Unless otherwise specified, we adopt neural networks for adjacency function ϕ and similarity function φ. Besides, we experimentally set $\lambda = 1$ to guide the learning of the adjacency function, providing reasonable adjacency scores for the relational graph.

3) Evaluation Protocols: On both datasets, we conduct 5-way 1-shot and 5-shot experiments which are standard few-shot learning settings. For evaluation, each episode is formed by randomly sampling 1 query for each of 5 classes. We report the mean accuracy (%) of 10 K randomly generated episodes as well as the 95% intervals on the test set.

B. Comparison With State-of-the-Arts

We first investigate the performance of our model, compared with state-of-the-art few-shot approaches, respectively, on *mini*ImageNet and *tiered*ImageNet. Since the representative ability of the embedding network has a significant impact on few-shot classification performance, we use four widely used architectures as embedding networks: Conv-4 [21], ResNet-12, ResNet-18 [58], and wide ResNet (WRN)-28 [59]. Moreover, we provide the classification performance of our TRPN-D-4 under three adjacency functions: Negative squared Euclidean

TABLE I
FEW-SHOT IMAGE CLASSIFICATION ACCURACIES OF 5-WAY 1-SHOT AND 5-WAY 5-SHOT TASKS ON *Mini*IMAGENET

Models	1-shot	5-shot
Conv-4		
MAML [45]	48.70 ± 1.84	63.10 ± 0.92
REPTILE [30]	49.97 ± 0.32	65.99 ± 0.58
GNN [19]	50.33 ± 0.36	66.41 ± 0.63
Matching Net [16]	43.56 ± 0.84	55.31 ± 0.73
Prototypical Net [17]	49.42± 0.78	68.20 ± 0.66
Relation Net [38]	50.40 ± 0.80	65.30 ± 0.70
RAPNETS [13]	-	70.89 ± 0.64
MPM [12]	57.59	74.02
EGNN [21]	-	76.37 ± 0.30
TRPN-T [25]	57.84 ± 0.51	78.57 ± 0.44
TRPN-D-4 + NE	58.50 ± 0.48	76.52 ± 0.44
TRPN-D-4 + Cosine	62.45 ± 0.50	80.45 ± 0.43
TRPN-D-4 + NN	**62.98 ± 0.50**	**81.24 ± 0.42**
ResNet-12		
SNAIL [29]	55.71 ± 0.99	68.88 ± 0.92
MTL [46]	61.20 ± 1.80	75.50 ± 0.80
TADAM [47]	58.50 ± 0.30	76.70 ± 0.30
TapNet [48]	61.65 ± 0.15	76.36 ± 0.10
LwoF [34]	55.45 ± 0.89	70.13 ± 0.68
TPN [20]	59.46	75.65
FSGC [15]	58.52±0.82	78.00±0.61
TRPN-T [25]	60.45 ± 0.53	81.05 ± 0.44
TRPN-D-4 + NE	66.30 ± 0.47	80.25 ± 0.47
TRPN-D-4 + Cosine	67.91 ± 0.48	84.42 ± 0.40
TRPN-D-4 + NN	**68.65 ± 0.48**	**84.69 ± 0.39**
ResNet-18		
Δ-encoder [49]	59.9	69.7
CTM [23]	62.05 ± 0.55	78.63 ± 0.06
AFHN [50]	62.38 ± 0.72	78.16 ± 0.56
Su et al. [51]	-	76.6 ± 0.7
TRPN-T [25]	56.70 ± 0.52	76.74 ± 0.46
TRPN-D-4 + NE	**68.93 ± 0.47**	85.45 ± 0.38
TRPN-D-4 + Cosine	68.00 ± 0.47	**86.05 ± 0.39**
TRPN-D-4 + NN	68.06 ± 0.47	84.50 ± 0.39
WRN-28		
LEO [24]	61.76 ± 0.08	77.59 ± 0.12
Param_Predict [35]	59.60 ± 0.41	73.74 ± 0.19
wDAE [36]	61.07 ± 0.15	76.75 ± 0.11
MPM [12]	61.77	78.03
IPN [11]	67.42 ± 0.45	83.98 ± 0.35
TRPN-T [25]	68.25 ± 0.50	85.40 ± 0.39
TRPN-D-4 + NE	69.73 ± 0.48	84.84 ± 0.39
TRPN-D-4 + Cosine	69.97 ± 0.48	85.18 ± 0.39
TRPN-D-4 + NN	**70.44 ± 0.48**	**85.43 ± 0.38**

TABLE II
FEW-SHOT IMAGE CLASSIFICATION ACCURACIES OF 5-WAY 1-SHOT AND 5-WAY 5-SHOT TASKS ON *Tiered*IMAGENET

Models	1-shot	5-shot
Conv-4		
MAML [45]	51.67 ± 1.81	70.30 ± 0.08
REPTILE [30]	52.36 ± 0.23	71.03 ± 0.22
GNN [19]	43.56 ± 0.84	55.31 ± 0.73
Matching Net [16]	54.02 ± 0.00	70.11 ± 0.00
Prototypical Net [17]	53.31 ± 0.89	72.69 ± 0.74
Relation Net [38]	54.48 ± 0.93	71.32 ± 0.70
LwoF [34]	50.90 ± 0.46	66.69 ± 0.36
TPN [20]	57.53 ± 0.96	72.85 ± 0.74
MPM [12]	59.44	76.59
EGNN [21]	-	80.15 ± 0.30
TRPN-T [25]	59.26 ± 0.50	79.66 ± 0.45
TRPN-D-4 + NE	56.60 ± 0.48	77.13 ± 0.44
TRPN-D-4 + Cosine	**62.88 ± 0.50**	80.57 ± 0.43
TRPN-D-4 + NN	61.01 ± 0.49	**80.98 ± 0.42**
ResNet-12		
TapNet [48]	63.08 ± 0.15	80.26 ± 0.12
DSN-MR [41]	67.39 ± 0.82	82.85 ± 0.56
E^3BM [55]	71.2 ± 0.4	85.3 ± 0.3
DeepEMD [56]	71.16 ± 0.87	86.03 ± 0.58
TRPN-T [25]	67.11 ± 0.50	83.07 ± 0.42
TRPN-D-4 + NE	68.80 ± 0.47	83.36 ± 0.39
TRPN-D-4 + Cosine	70.58 ± 0.47	86.06 ± 0.37
TRPN-D-4 + NN	**71.45 ± 0.47**	**87.13 ± 0.37**
ResNet-18		
CTM [23]	64.78 ± 0.11	81.05 ± 0.13
Su et al. [51]	-	78.9 ± 0.7
TRPN-T [25]	67.01 ± 0.48	83.60 ± 0.48
TRPN-D-4 + NE	70.89 ± 0.47	88.70 ± 0.35
TRPN-D-4 + Cosine	69.56 ± 0.48	**88.67 ± 0.35**
TRPN-D-4 + NN	**72.47 ± 0.47**	87.84 ± 0.36
WRN-28		
LEO [24]	66.33 ± 0.05	81.44 ± 0.09
wDAE [36]	68.18 ± 0.16	83.09 ± 0.12
MPM [12]	67.58	83.93
Centroid alignment [57]	74.40 ± 0.68	86.61 ± 0.59
IPN [11]	73.18 ± 0.43	86.59 ± 0.33
TRPN-T [25]	70.25 ± 0.50	85.21 ± 0.37
TRPN-D-4 + NE	71.37 ± 0.47	85.51 ± 0.38
TRPN-D-4 + Cosine	**73.43 ± 0.47**	**87.48 ± 0.36**
TRPN-D-4 + NN	73.31 ± 0.47	87.42 ± 0.36

distance (namely NE), Cosine Similarity (namely Cosine), and Neural Network (namely NN).

Tables I and II list the few-shot classification accuracies of the 5-way 1-shot and 5-shot tasks along with the specifications of the embedding models for feature extraction. From both Tables I and II, we can find that TRPN-D-4 + NN has a stable and good performance with all embedding networks. Nonparametric cosine similarity can also achieve better performance when the backbone is deep such as WRN-28, due to the strong representation ability of the backbone. Negative squared Euclidean distance performs worse than both Neural network and Cosine Similarity. To better compared our TRPN-D with other methods, TRPN-D below all refers to TRPN-D + NN.

From Table I, we can observe that the proposed TRPN-D achieves superior performance among the state-of-the-art few-shot approaches with all backbone architectures, proving the effectiveness and robustness of our method. Moreover, the proposed TRPN-D beats TRPN-T by significant margins in most cases, which validates our claim that decoupling training strategy could help bridge the semantic gap and empower the few-shot model the ability of transferring knowledge to new tasks. It can be shown that, on *mini*Imagenet dataset, the performance of the proposed TRPN-D with WRN-28 is nearly close to that of TRPN-T with only 0.03% gains, while the performance gains of more complex scenes, such as 1-shot tasks, or tasks on more challenging dataset *tired*Imagenet,

are conspicuous. That is to say, our approach is more suitable for realistic complicated scenarios. Besides, results of TRPN-T usually rank the highest except TRPN-D. Meanwhile, TRPN with "Conv-4" backbone achieves 57.84% and 78.57%, respectively, on 1-shot and 5-shot settings, even outperforming most of the state-of-the-art models equipped with deeper embedding networks. These phenomena confirm the improvement brought by the relation propagation.

A similar trend can also be observed in Table II. The proposed TRPN-D shows comparable results with the state of the art, achieving 72.31% and 87.19% accuracy on 5-way 1-shot and 5-shot setting, respectively. It delivers obvious gains to TRPN-T. As the shots increase, all the methods perform better, which is adhere to our intuition. Moreover, a deeper embedding network will lead to a better classification performance compared to methods equipped with Conv4. However, the performance of the model with ResNet-12 is similar to that with ResNet-18.

From these observations, we can conclude that our TRPN-D adequately exploits the support-query pair relations and improves the ability to transfer knowledge to new tasks, finally achieving decent few-shot classification performance.

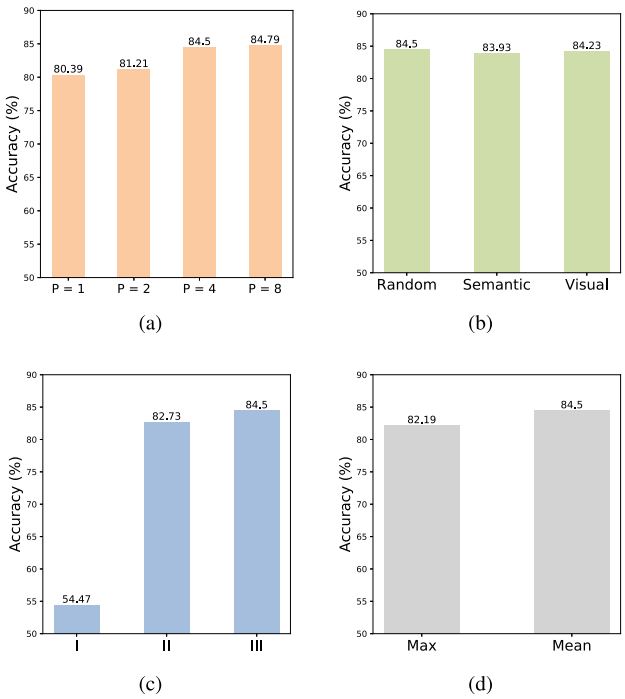

Fig. 3. Effects of Different Decoupling Implementation of 5-way 5-shot task on *mini*Imagenet. (a) Effects of the number of data partitions. (b) Effects of different class-selection mechanisms. (c) Effects of different task-construction approaches. (d) Effects of different weighted voting approaches.

C. Effects of Different Decoupling Implementation

There are different implementations for the proposed decoupling strategy, including the number of partitions and class-selection mechanism. We evaluate the effect of each implementation and determine the best combination.

1) Number of Data Partitions: In the proposed decoupling training, the dataset is grouped into P groups. That is to say, there should be P parallel embedding networks to be trained using P data groups. However, the more embedding networks are involved, the more computation resources are required. Therefore, we conduct experiments to find out the appropriate value of P, compromising the efficiency and accuracy. As shown in Fig. 3(a), the x-coordinate lists the few-shot classification accuracy of TRPN-D models with $P = 1, 2, 4, 8$ embedding networks, respectively. It can be observed that the performance gains as the number of embedding networks increases. And, the increasing speed of performance gains becomes slow when $P = 4$. Specifically, there is only 0.2% gains from TRPN-D-4 to TRPN-D-8. Hence, in our experiment, we acquiescently set $P = 4$.

2) Class-Selection Mechanism: We consider three selection mechanisms to group different classes: 1) random-based mechanism. Each class is randomly picked for a group, and the classes are balanced in each group. 2) semantic-based mechanism. Similar semantic word2vec embeddings of training class labels are grouped into the same partition via K-means clustering method. We set the number of clusters to P, and employ a hyperheuristic algorithm based on random selection to continuously adjust the initial cluster centers of K-means until the maximum number of iterations is reached. 3) visual-based mechanism. The visual-based method is similar to the semantic-based method, except that it used the average visual embedding extracted by the pretrained embedding network of each category in the training set of *mini*ImageNet instead of the word2vec embedding. As we can see from Fig. 3(b), the results of the three mechanisms are very close, from nearly 84% to 84.5%. Intuitively, it seems that a mechanism based on visual or semantic is more reasonable for training a few-shot module since dissimilar clusters could better simulate the few-shot experimental setting where categories of test dataset are dissimilar from that of train dataset. However, using similar samples in a cluster with limited diversity will do harm to the representative ability of the embedding network. The impact of diverse training data for the embedding network could also be seen in **Task-construction approach**. Therefore, the mechanisms based on visual or semantic do not achieve superior performance as we expect. Besides, the result of the random-based mechanism ranks the highest among the three mechanisms. Therefore, we simply choose the random-based mechanism in our experiments.

3) Task-Construction Approach: The proposed decoupling training strategy is using different data to train the embedding networks and TRPNs to imitate the test scenario, which helps TRPNs to capture and propagate unseen relations. As we introduced in Section III-D, the training classes are divided into P disjoint partitions and each (P-1) partitions form a group. Different division methods will cause different task construction approaches. Actually, we have tried three task-construction methods:

1) Using samples of different partitions $\mathcal{C}_{\text{train}}^{(p)}$ to train the embedding networks $\mathcal{F}_{(p)}$, and using the other partitions to train $\mathcal{G}_{(p)}$

2) Using different data groups $\mathcal{D}_{(p)}$ to train $\mathcal{F}_{(p)}$, and the corresponding samples of different partitions $\mathcal{C}_{\text{train}}^{(p)}$ to train $\mathcal{G}_{(p)}$
3) Using different data groups $\mathcal{D}_{(p)}$ to train $\mathcal{F}_{(p)}$, and the whole dataset $\mathcal{D}_{\text{train}}$ to train $\mathcal{G}_{(p)}$

As Fig. 3(c) illustrates, the first method only adopts a comparatively few classes to train the embedding networks, leading to the inferior representative ability of the backbone. Therefore, feature maps extracted from the rest unseen samples by the pretrained embedding networks contain a lot of noise so that the relational propagation graph network can't capture valid relation information and thus greatly hurt the classification accuracy.

The second method ensures the splendid feature distillation of the embedding networks, and the completely unseen samples used to train the TRPN module seem to better learn the transfer ability. However, the completely unseen partitions result in fewer unseen few-shot tasks and the diversity of seen samples is invisible to the GNN. As a result, the few-shot classification performance of the second method is worse than that of the third method.

The third method, i.e., our choice, ensures the TRPN to better extract the relations on new tasks unseen to the feature embedding network, and in the meanwhile maintains the discriminative representation ability of the feature embedding module, achieving the best performance compared to the others.

4) Weighted Voting Approach: Actually, we testify two kinds of weighted voting approach of the final results in the test phase, namely average weighted voting and maximum decisive voting. The formal definitions are listed as follows:

1) average weighted voting

$$\overline{\mathbf{S}}_{qn} = \frac{1}{P}\sum_{p=1}^{P}\sum_{\{i|(\mathbf{x}_i,y_i)\in\mathcal{S}\wedge y_i=\mathcal{C}_n\}}\mathbf{S}_{qi}^{(p)}$$

2) maximum weighted voting

$$\overline{\mathbf{S}}_{qn} = \max_{p=1,\ldots,P}\left(\sum_{\{i|(\mathbf{x}_i,y_i)\in\mathcal{S}\wedge y_i=\mathcal{C}_n\}}\mathbf{S}_{qi}^{(p)}\right).$$

Fig. 3(d) reports the comparison between the performance of the two approaches, from which it is obvious that average weighted voting achieves better performance by a large margin, nearly 2.5% compared to the counterpart.

According to that analysis, our TRPN-D adopts decoupling training strategy integrating above choices and yields peak performance.

D. Component Analysis

As aforementioned, our method mainly gains from the relational graph framework with decoupling training. Here, we study the effects of different components in our TRPN-D model. To clearly demonstrate the effects of each part of the proposed few-shot module TRPN, we first conduct experiments on TRPN-D-1. The first five bars in Fig. 4 demonstrate the few-shot classification results of TRPN-D-1 without relational nodes where we construct graph node with a single sample and its corresponding one-hot label, under nontransductive setting, without pseudo relational nodes, without interclass uniqueness in adjacency matrix where we directly use the adjacency function to evaluate the adjacency between two samples from different classes, and that obtained by the full TRPN-D-1 framework. Note that there is no decoupling training for TRPN-D-1, since the model only contains one embedding network. In the meanwhile, the last bar in Fig. 4 indicates the result of TRPN-D-4.

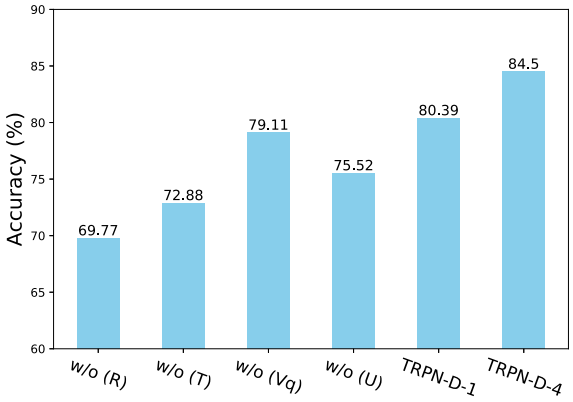

Fig. 4. Ablation study of 5-way 5-shot task on *mini*ImageNet. It, respectively, shows the few-shot classification results of TRPN-D-1 without relational nodes (w/o(R)), under nontransductive setting (w/o(T)), without pseudo relational nodes (w/o(Vq)), without interclass uniqueness in adjacency matrix (w/o(U)), the results of the full TRPN-D-1 (TRPN-D-1), and results acquired by the TRPN-D-4 (TRPN-D-4) from left to right.

From Fig. 4, we can see that both the TRPN-D-1 without relational nodes and that under nontransductive setting cannot utilize sufficient information existing in the task and thus suffers from a significant performance drop. Without relational nodes, the performance drops more than 10%, which greatly manifests the importance of the support-query relations. TRPN-D-1 under the transductive setting achieves better performance than TRPN-D-1 under the nontransductive setting (80.3% versus 72.88%). Actually, a similar conclusion can be also drawn from the experiments with TRPN-D-4 (84.5% versus 76.56%), which we do not show in the figure to avoid redundancy. Without pseudo relational nodes, the TRPN-D-1 witnesses a drop from 80.39% to 79.11%, declining by over 1%. Without considering interclass uniqueness, the results decrease by 5% around, due to neglecting the commonality of different categories. Although TRPN-D-1 achieves competitive performance, it falls far short of TRPN-D-4, demonstrating that the proposed decoupling training strategy is capable of transferring information and accomplishing new tasks.

E. Propagation Visualization

To further demonstrate how our relation-propagation graph network propagates the relations of the support-query pairs, we, respectively, visualize the relations including the initial coarse adjacency based on the raw node features, the predicted

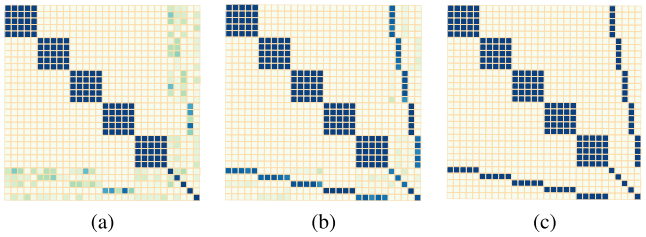

Fig. 5. Visualization of the relation propagation. (a) Initial adjacency. (b) Similarity score after TRPN-D modeling and propagating the relations. (c) Ground-truth.

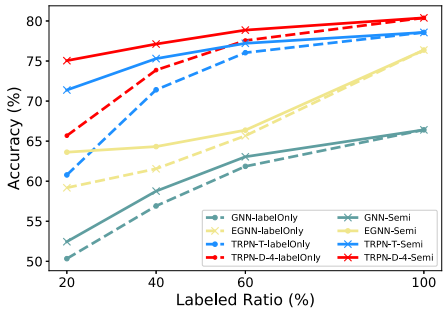

Fig. 6. Semisupervised few-shot classification accuracies of the 5-way 5-shot tasks on *mini*ImageNet.

similarities based on the learned relational embeddings and the ground-truth in Fig. 5. Since we mainly focus on support-query relations which we marked with red bound boxes in Fig. 5, the relations between support samples are simply visualized according to the given labels. From Fig. 5, we can draw the conclusion that starting from the coarse prediction between support-query pairs where relations are uncertain with some randomness, our TRPN module can precisely predict the true relations using the relation propagation, with the help of the decoupling training strategy which enhances the ability of transferring past knowledge to a new task.

F. Semisupervised Few-Shot Classification

For showing more efficient relation propagation on the graph, we extend our graph-based architecture to learn from a mixture of labeled and unlabeled examples. We follow the same semisupervised settings proposed in [19], [21] for a fair comparison. Since the two works did not provide the results around 60% labeled ratio and 40% labeled ratio, we retrained the two models with their published codes. Specifically, two settings are considered in this experiment, namely "Semi" and "LabeledOnly". "Semi" means a 5-way 5-shot setting with partially labeled samples in the support set where labeled samples are balanced among classes, while "LabeledOnly" is based on "Semi" and denotes learning with only labeled support samples. We compare four models equipped with the "Conv-4" backbone including GNN, edge enhanced GNN (EGNN), TRPN-T, and TRPN-D. As shown in Fig. 6, the TRPN-T shows a better learning ability as a labeled ratio increases especially on "LabeledOnly" setting, verifying that it could learn the discriminative relation embeddings from the given labels. With the decoupling strategy, TRPN-D surpasses other graph settings which prove that our method is effective even under semisupervised settings. Models under "Semi" setting usually perform better than that under "LabeledOnly" setting, proving the importance of the samples themselves even without semantic labels.

V. Conclusion

In this article, we presented a novel TRPN-D, the first work which explicitly decouples the embedding network and the few-shot module with different tasks, guaranteeing that the few-shot module TRPN gains the ability to transfer knowledge in order to adapt to new tasks. Directly modeling and propagating the relations across support-query pairs further help capture the accurate relations among the tasks. Extensive experiments conducted on several benchmark datasets demonstrate the superiority of TRPN compared with state-of-the-art few-shot learning methods. With a decoupling training strategy, the proposed TRPN-D consistently brings significant gains in several experimental settings, compared to TRPN. We also carry out experiments to determine the appropriate implementation for the decoupling strategy, ensuring that our implementation could achieve peak performance. It can be concluded that our TRPN-D is capable of effectively capturing the relations and adapting to new tasks quickly.

References

[1] L. Shao, D. Wu, and X. Li, "Learning deep and wide: A spectral method for learning deep networks," *IEEE Trans. Neural Netw. Learn. Syst.*, vol. 25, no. 12, pp. 2303–2308, Dec. 2014.

[2] D. Gong, Z. Zhang, Q. Shi, A. van den Hengel, C. Shen, and Y. Zhang, "Learning deep gradient descent optimization for image deconvolution," *IEEE Trans. Neural Netw. Learn. Syst.*, vol. 31, no. 12, pp. 5468–5482, Feb. 2020.

[3] G. M. Park, S. M. Yoo, and J. H. Kim, "Convolutional neural network with developmental memory for continual learning," *IEEE Trans. Neural Netw. Learn. Syst.*, vol. 32, no. 6, pp. 2691–2705, Jun. 2020.

[4] L. Shao, F. Zhu, and X. Li, "Transfer learning for visual categorization: A survey," *IEEE Trans. Neural Netw. Learn. Syst.*, vol. 26, no. 5, pp. 1019–1034, May 2015.

[5] A. Krizhevsky, I. Sutskever, and G. E. Hinton, "ImageNet classification with deep convolutional neural networks," in *Proc. Adv. Neural Inf. Process. Syst.*, 2012, pp. 1097–1105.

[6] L.-C. Chen, G. Papandreou, I. Kokkinos, K. Murphy, and A. L. Yuille, "DeepLab: Semantic image segmentation with deep convolutional nets, atrous convolution, and fully connected CRFs," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 40, no. 4, pp. 834–848, Apr. 2018.

[7] X. Chen, J. Weng, W. Lu, J. Xu, and J. Weng, "Deep manifold learning combined with convolutional neural networks for action recognition," *IEEE Trans. Neural Netw. Learn. Syst.*, vol. 29, no. 9, pp. 3938–3952, Sep. 2018.

[8] Z.-Q. Zhao, P. Zheng, S.-T. Xu, and X. Wu, "Object detection with deep learning: A review," *IEEE Trans. Neural Netw. Learn. Syst.*, vol. 30, no. 11, pp. 3212–3232, Nov. 2019.

[9] L. Fei-Fei, R. Fergus, and P. Perona, "One-shot learning of object categories," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 28, no. 4, pp. 594–611, Apr. 2006.

[10] B. Lake, R. Salakhutdinov, J. Gross, and J. Tenenbaum, "One shot learning of simple visual concepts," in *Proc. Annu. Meeting Cognit. Sci. Soc.*, vol. 33, no. 33, pp. 2568–2573, 2011.

[11] Y. Ma *et al.*, "Few-shot visual learning with contextual memory and fine-grained calibration," in *Proc. 29th Int. Joint Conf. Artif. Intell.*, Jul. 2020, pp. 811–817.

[12] N. Lai, M. Kan, C. Han, X. Song, and S. Shan, "Learning to learn adaptive classifier-predictor for few-shot learning," *IEEE Trans. Neural Netw. Learn. Syst.*, early access, 2020, doi: 10.1109/TNNLS.2020.3011526.

[13] J. Lu, S. Jin, J. Liang, and C. Zhang, "Robust few-shot learning for user-provided data," *IEEE Trans. Neural Netw. Learn. Syst.*, early access, doi: 10.1109/TNNLS.2020.2984710.

[14] H. Huang, J. Zhang, J. Zhang, J. Xu, and Q. Wu, "Low-rank pairwise alignment bilinear network for few-shot fine-grained image classification," *IEEE Trans. Multimedia*, vol. 23, pp. 1666–1680, 2021.

[15] H.-G. Jung and S.-W. Lee, "Few-shot learning with geometric constraints," *IEEE Trans. Neural Netw. Learn. Syst.*, vol. 31, no. 11, pp. 4660–4672, Nov. 2020.

[16] O. Vinyals, C. Blundell, T. Lillicrap, K. Kavukcuoglu, and D. Wierstra, "Matching networks for one shot learning," in *Proc. Adv. Neural Inf. Process. Syst.*, 2016, pp. 3630–3638.

[17] J. Wang and Y. Zhai, "Prototypical siamese networks for few-shot learning," in *Proc. IEEE 10th Int. Conf. Electron. Inf. Emergency Commun. (ICEIEC)*, Jul. 2020, pp. 4077–4087.

[18] W. Li, J. Xu, J. Huo, L. Wang, Y. Gao, and J. Luo, "Distribution consistency based covariance metric networks for few-shot learning," in *Proc. AAAI Conf. Artif. Intell.*, vol. 33, Jul. 2019, pp. 8642–8649.

[19] V. G. Satorras and J. B. Estrach, "Few-shot learning with graph neural networks," in *Proc. Int. Conf. Learn. Represent.*, 2018, pp. 1–13.

[20] Y. Liu *et al.*, "Learning to propagate labels: Transductive propagation network for few-shot learning," in *Proc. 7th Int. Conf. Learn. Represent.*, 2019, pp. 1–14.

[21] J. Kim, T. Kim, S. Kim, and C. D. Yoo, "Edge-labeling graph neural network for few-shot learning," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2019, pp. 11–20.

[22] H.-J. Ye, H. Hu, D.-C. Zhan, and F. Sha, "Few-shot learning via embedding adaptation with set-to-set functions," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2020, pp. 8808–8817.

[23] H. Li, D. Eigen, S. Dodge, M. Zeiler, and X. Wang, "Finding task-relevant features for few-shot learning by category traversal," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2019, pp. 1–10.

[24] A. A. Rusu *et al.*, "Meta-learning with latent embedding optimization," in *Proc. Int. Conf. Learn. Represent.*, 2018, pp. 1–17.

[25] Y. Ma, S. Bai, S. An, W. Liu, and X. Liu, "Transductive relation-propagation network for few-shot learning," in *Proc. 29th Int. Joint Conf. Artif. Intell. 17th Pacific Rim Int. Conf. Artif. Intell. (IJCAI-PRICAI)*, 2020, pp. 804–810.

[26] M. Ren *et al.*, "Meta-learning for semi-supervised few-shot classification," 2018, *arXiv:1803.00676*. [Online]. Available: https://arxiv.org/abs/1803.00676

[27] A. Santoro, S. Bartunov, M. Botvinick, D. Wierstra, and T. Lillicrap, "Meta-learning with memory-augmented neural networks," in *Proc. Int. Conf. Mach. Learn.*, 2016, pp. 1842–1850.

[28] S. Ravi and H. Larochelle, "Optimization as a model for few-shot learning," in *Proc. Int. Conf. Learn. Represent.*, 2017, pp. 1–11.

[29] N. Mishra, M. Rohaninejad, X. Chen, and P. Abbeel, "A simple neural attentive meta-learner," 2017, *arXiv:1707.03141*. [Online]. Available: http://arxiv.org/abs/1707.03141

[30] A. Nichol, J. Achiam, and J. Schulman, "On first-order meta-learning algorithms," 2018, *arXiv:1803.02999*. [Online]. Available: http://arxiv.org/abs/1803.02999

[31] S. Flennerhag, A. A. Rusu, R. Pascanu, F. Visin, H. Yin, and R. Hadsell, "Meta-learning with warped gradient descent," 2019, *arXiv:1909.00025*. [Online]. Available: http://arxiv.org/abs/1909.00025

[32] T. Elsken, B. Staffler, J. H. Metzen, and F. Hutter, "Meta-learning of neural architectures for few-shot learning," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2020, pp. 12365–12375.

[33] Y.-X. Wang, R. Girshick, M. Hebert, and B. Hariharan, "Low-shot learning from imaginary data," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit.*, Jun. 2018, pp. 7278–7286.

[34] S. Gidaris and N. Komodakis, "Dynamic few-shot visual learning without forgetting," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit.*, Jun. 2018, pp. 4367–4375.

[35] S. Qiao, C. Liu, W. Shen, and A. Yuille, "Few-shot image recognition by predicting parameters from activations," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit.*, Jun. 2018, pp. 7229–7238.

[36] S. Gidaris and N. Komodakis, "Generating classification weights with GNN denoising autoencoders for few-shot learning," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2019, pp. 21–30.

[37] Y. Guo and N.-M. Cheung, "Attentive weights generation for few shot learning via information maximization," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2020, pp. 13499–13508.

[38] F. Sung, Y. Yang, L. Zhang, T. Xiang, P. H. S. Torr, and T. M. Hospedales, "Learning to compare: Relation network for few-shot learning," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit.*, Jun. 2018, pp. 1199–1208.

[39] W. Li, L. Wang, J. Xu, J. Huo, Y. Gao, and J. Luo, "Revisiting local descriptor based image-to-class measure for few-shot learning," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2019, pp. 7260–7268.

[40] A. Li, W. Huang, X. Lan, J. Feng, Z. Li, and L. Wang, "Boosting few-shot learning with adaptive margin loss," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit.*, 2020, pp. 12576–12584.

[41] C. Simon, P. Koniusz, R. Nock, and M. Harandi, "Adaptive subspaces for few-shot learning," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2020, pp. 4136–4145.

[42] L. Yang, L. Li, Z. Zhang, X. Zhou, E. Zhou, and Y. Liu, "DPGN: Distribution propagation graph network for few-shot learning," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2020, pp. 13390–13399.

[43] Q. Li, Z. Han, and X.-M. Wu, "Deeper insights into graph convolutional networks for semi-supervised learning," in *Proc. 32nd AAAI Conf. Artif. Intell.*, 2018, pp. 3538–3545.

[44] M. Lichtenstein, P. Sattigeri, R. Feris, R. Giryes, and L. Karlinsky, "TAFSSL: Task-adaptive feature sub-space learning for few-shot classification," 2020, *arXiv:2003.06670*. [Online]. Available: http://arxiv.org/abs/2003.06670

[45] C. Finn, P. Abbeel, and S. Levine, "Model-agnostic meta-learning for fast adaptation of deep networks," in *Proc. 34th Int. Conf. Mach. Learn.*, 2017, pp. 1126–1135.

[46] Q. Sun, Y. Liu, T.-S. Chua, and B. Schiele, "Meta-transfer learning for few-shot learning," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2019, pp. 403–412.

[47] B. Oreshkin, P. R. López, and A. Lacoste, "TADAM: Task dependent adaptive metric for improved few-shot learning," in *Proc. Adv. Neural Inf. Process. Syst.*, 2018, pp. 721–731.

[48] S. W. Yoon, J. Seo, and J. Moon, "TapNet: Neural network augmented with task-adaptive projection for few-shot learning," in *Proc. Int. Conf. Mach. Learn.*, 2019, pp. 7115–7123.

[49] E. Schwartz *et al.*, "Delta-encoder: An effective sample synthesis method for few-shot object recognition," in *Proc. Adv. Neural Inf. Process. Syst.*, 2018, pp. 2845–2855.

[50] K. Li, Y. Zhang, K. Li, and Y. Fu, "Adversarial feature hallucination networks for few-shot learning," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2020, pp. 13470–13479.

[51] J.-C. Su, S. Maji, and B. Hariharan, "When does self-supervision improve few-shot learning?" in *Proc. Eur. Conf. Comput. Vis.*, 2020, pp. 645–666.

[52] H. Qi, M. Brown, and D. G. Lowe, "Low-shot learning with imprinted weights," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit.*, Jun. 2018, pp. 5822–5830.

[53] S. Gidaris, A. Bursuc, N. Komodakis, P. P. Pérez, and M. Cord, "Boosting few-shot visual learning with self-supervision," in *Proc. IEEE/CVF Int. Conf. Comput. Vis. (ICCV)*, Oct. 2019, pp. 8059–8068.

[54] D. P. Kingma and J. Ba, "Adam: A method for stochastic optimization," 2014, *arXiv:1412.6980*. [Online]. Available: http://arxiv.org/abs/1412.6980

[55] Y. Liu, B. Schiele, and Q. Sun, "An ensemble of epoch-wise empirical Bayes for few-shot learning," in *Proc. Eur. Conf. Comput. Vis.*, 2020, pp. 404–421.

[56] C. Zhang, Y. Cai, G. Lin, and C. Shen, "DeepEMD: Few-shot image classification with differentiable earth mover's distance and structured classifiers," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit.*, Jun. 2020, pp. 12203–12213.

[57] A. Afrasiyabi, J.-F. Lalonde, and C. Gagné, "Associative alignment for few-shot image classification," in *Proc. Eur. Conf. Comput. Vis.*, Aug. 2020, pp. 18–35.

[58] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2016, pp. 770–778.

[59] S. Zagoruyko and N. Komodakis, "Wide residual networks," 2016, *arXiv:1605.07146*. [Online]. Available: http://arxiv.org/abs/1605.07146

Yuqing Ma received the B.S. degree from Shandong University, Jinan, China, in 2015. She is currently pursuing the Ph.D. degree with the School of Computer Science and Engineering, Beihang University, Beijing, China.

Her current research interests include computer vision, generative models, and few-shot learning.

Shihao Bai received the B.E. degree from Beihang University, Beijing, China, in 2020, where he is currently pursuing the M.A. degree with the School of Computer Science and Engineering.

His research interests include machine learning algorithms, computer vision, and few-shot learning.

Wei Liu received the B.E. degree from Beihang University, Beijing, China, in 2020.

He is currently an Intern with the State Key Laboratory of Software Development Environment, Beihang University, His research interests include computer vision and few-shot learning.

Shuo Wang received the B.E. degree from Beihang University, Beijing, China, in 2019, where she is currently pursuing the master's degree with the School of Computer Science and Engineer.

Her research interests include computer vision and few-shot learning.

Yue Yu received the Ph.D. degree in computer science from National University of Defense Technology (NUDT), Changsha, China, in 2016.

He is an Associate Professor with the College of Computer, NUDT. He has won Outstanding Ph.D. Thesis Award from Hunan Province. His research findings have been published on the International Conference on Software Engineering (ICSE), the Foundations of Software Engineering (FSE), ASE, Empirical Software Engineering and Measurement (ESEM) etc. His current research interests include software engineering, data mining and computer supported cooperative work.

Xiao Bai received the B.Eng. degree in computer science from Beihang University, Beijing, China, in 2001, and the Ph.D. degree from the University of York, York, U.K., in 2006.

He was a Research Officer (Fellow, Scientist) with the Computer Science Department, University of Bath, Bath, U.K., until 2008. He is currently an Full Professor with the School of Computer Science and Engineering, Beihang University. He has published more than 60 articles in journals and refereed conferences. His current research interests include pattern recognition, image processing, and remote sensing image analysis.

Dr. Bai has been awarded the New Century Excellent Talents in University by the Ministry of Education of China in 2012 and the Excellency Scientist by the China Association for Science and Technology in 2016. He is also a Guest Editor of the *Pattern Recognition* Special Issue on *Pattern Recognition for High Performance Imaging*.

Xianglong Liu (Member, IEEE) received the B.S. and Ph.D. degrees in computer science from Beihang University, Beijing, China, in 2008 and 2014, respectively.

From 2011 to 2012, he visited the Digital Video and Multimedia (DVMM) Lab, Columbia University as a joint Ph.D. student. He is currently an Associate Professor with the School of Computer Science and Engineering, Beihang University. He has published over 40 research articles at top venues like the IEEE TRANSACTIONS ON IMAGE PROCESSING, the IEEE TRANSACTIONS ON CYBERNETICS, IEEE TRANSACTIONS ON NEURAL NETWORKS AND LEARNING SYSTEMS, the Conference on Computer Vision and Pattern Recognition, the International Conference on Computer Vision, and the Association for the Advancement of Artificial Intelligence. His research interests include machine learning, computer vision, and multimedia information retrieval.

Meng Wang (Fellow, IEEE) received the B.E. and Ph.D. degrees in special class for the gifted young from the Department of Electronic Engineering and Information Science, University of Science and Technology of China (USTC), Hefei, China, in July 2003 and July 2008, respectively.

He is a Professor with the Hefei University of Technology, Hefei, China. He previously worked as an Associate Researcher with Microsoft Research Asia, Beijing, China, and then a Core Member in a startup in Silicon Valley. After that, he worked with the National University of Singapore, Singapore, as a Senior Research Fellow. He has authored more than 100 book chapters, journal, and conference papers in these areas. His current research interests include multimedia content analysis, search, mining, recommendation, and large-scale computing.

Dr. Wang is a member of the Association for Computing Machinery (ACM). He was a recipient of the Best Paper Awards continuously in the 17th and 18th ACM International Conference on Multimedia and the Best Paper Award in the 16th International Multimedia Modeling Conference.