Learning to Learn Adaptive Classifier–Predictor for Few-Shot Learning

Nan Lai, Meina Kan, *Member, IEEE*, Chunrui Han, Xingguang Song, and Shiguang Shan, *Senior Member, IEEE*

***Abstract*—Few-shot learning aims to learn a well-performing model from a few labeled examples. Recently, quite a few works propose to learn a predictor to directly generate model parameter weights with episodic training strategy of meta-learning and achieve fairly promising performance. However, the predictor in these works is task-agnostic, which means that the predictor cannot adjust to novel tasks in the testing phase. In this article, we propose a novel meta-learning method to learn how to learn task-adaptive classifier–predictor to generate classifier weights for few-shot classification. Specifically, a meta classifier-predictor module, (MPM) is introduced to learn how to adaptively update a task-agnostic classifier–predictor to a task-specialized one on a novel task with a newly proposed center-uniqueness loss function. Compared with previous works, our task-adaptive classifier–predictor can better capture characteristics of each category in a novel task and thus generate a more accurate and effective classifier. Our method is evaluated on two commonly used benchmarks for few-shot classification, i.e., miniImageNet and tieredImageNet. Ablation study verifies the necessity of learning task-adaptive classifier–predictor and the effectiveness of our newly proposed center-uniqueness loss. Moreover, our method achieves the state-of-the-art performance on both benchmarks, thus demonstrating its superiority.**

***Index Terms*—Few-shot learning, meta-learning, predict classifier weights, task-adaptive predictor.**

I. Introduction

IMPRESSIVE advances of deep learning in various tasks [1]–[3] depend on massive labeled data, which provides abundant information to support the learning of numerous parameters. However, collecting plenty of labeled data involves huge labor and cost and is even impossible in some areas, such as medical imaging and terrorist monitoring. Hence, equipping a deep model with the ability to learn new concepts from a few labeled data is meaningful for practical use. Recently, research topics that attempt to learn new concepts from limited amount of labeled data have attracted increasing attention, such as zero-shot learning [4]–[6], few-shot learning [7], [8], and continual learning [9], [10]. Facing the same challenge of limited data, these research topics target different tasks under different settings. Zero-shot learning aims to recognize new concepts with no labeled data of new concepts provided and few-shot learning devotes to recognize new concepts from several labeled data of these concepts, while continual learning focuses on learning new concepts without forgetting concepts learned before. This work focuses on few-shot learning.

Few-shot learning is such a research topic that studies how to learn a new concept from few training data of this concept and has received significant attention from the machine learning community. Although humans can easily learn new concepts from very few examples, it remains a big challenge for machine learning approaches. In the early stage, generative approaches [11]–[13] propose to model the composition of objects from parts using probabilistic graphical models. These methods work well on simple objects, e.g., handwritten digits, but can hardly handle categories with vast variations. By contrast, powerful deep neural networks can well deal with complex variations and have favorable transferability across different tasks [14], [15]. However, in the setting of few-shot learning, directly training a deep model or fine-tuning a pretrained deep model with few training data would cause severe overfitting problem.

Recently, casting few-shot learning as a meta-learning problem becomes prevalent and has achieved significant improvement. Some approaches [16]–[18] learn a well-generalized model across different few-shot tasks, and the achieved model is directly applied to novel tasks without model adaptation. Besides, some optimization-based meta-learning approaches [7], [19]–[22] aim at a task-specialized model by learning to optimize model parameters with few samples from a novel task. Lately, instead of optimizing model parameters with few shots, some works [23]–[26] directly learn to predict model parameters given only few samples of the task. These meta-learning approaches achieve more impressive performance than the early generative approaches,

Manuscript received November 12, 2019; revised May 23, 2020; accepted July 11, 2020. This work was supported in part by the National Key Research and Development Program of China under Grant 2017YFA0700800, in part by the Natural Science Foundation of China under Grant 61772496, and in part by the Youth Innovation Promotion Association of Chinese Academy of Sciences under Grant 2017145. *(Corresponding author: Meina Kan.)*

Nan Lai, Meina Kan, and Chunrui Han are with the Key Laboratory of Intelligent Information Processing, Institute of Computing Technology (ICT), Chinese Academy of Sciences (CAS), Beijing 100190, China, and also with the University of Chinese Academy of Sciences, Beijing 100049, China (e-mail: lainan@ict.ac.cn; kanmeina@ict.ac.cn; chunrui.han@vipl.ict.ac.cn).

Xingguang Song is with Huawei Technologies Company Ltd., Shenzhen 518129, China (e-mail: songxingguang@huawei.com).

Shiguang Shan is with the Key Laboratory of Intelligent Information Processing, Institute of Computing Technology (ICT), Chinese Academy of Sciences (CAS), Beijing 100190, China, also with the University of Chinese Academy of Sciences, Beijing 100049, China, and also with the CAS Center for Excellence in Brain Science and Intelligence Technology, Shanghai 200031, China (e-mail: sgshan@ict.ac.cn).

Color versions of one or more of the figures in this article are available online at http://ieeexplore.ieee.org.

Digital Object Identifier 10.1109/TNNLS.2020.3011526

suggesting that meta-learning is a very promising approach to few-shot learning for its essence of learning to learn. What to learn, i.e., the designing of meta-learner, is the vital key and is also the major difference between various meta-learning methods.

A few latest meta-learning works [23], [24], [26], [27] dedicate to learning a parameter predictor to directly generate model weights for few-shot classification. For example, Li *et al.* [26] introduced a MetaNet module to generate the parameter weights of feature encoder to extract more discriminative feature representation. Their promising performance demonstrates that directly predicting weights of the target model via a predictor is a superior way to few-shot classification. However, the parameter predictor in these works keeps fixed in the testing stage and is shared by different novel tasks, i.e., the predictor is task-agnostic. Despite good generalization performance, the task-agnostic predictor cannot capture the peculiar characteristic of a novel task. Considering this, in this work, we propose a novel meta-learning method that learns how to learn task-adaptive parameter predictor given only few shots in order to predict more accurate parameter weights for a novel task.

Given that a task-adaptive parameter predictor should vary with the categories of a task, we model our task-adaptive classifier–predictor as a composition of multiple category-specific generators, each of which is specialized for one category in a novel task to generate the corresponding classifier weights. Specifically, we propose to update a universal generator to multiple category-specific generators to get task-adaptive classifier–predictor. For this purpose, a novel meta classifier-predictor module (MPM) is introduced to learn a good universal generator, which is further updated to multiple category-specific ones with only few shots through a new loss function named center-uniqueness loss. These category-specific generators better capture the specific variation of each category and thus are capable of generating more accurate classifier weights for a novel task.

Our method is evaluated on two commonly used benchmarks for few-shot classification problem, i.e., miniImageNet and tieredImageNet. Comprehensive ablation studying on both datasets shows the necessity of task-adaptive classifier–predictor and the effectiveness of our proposed MPM module and the center-uniqueness loss. In addition, our method achieves state-of-the-art performance on both two benchmarks.

II. Related Work

Few-shot learning has been studied for decades and considerable progress has been made up to now. Approaches for few-shot learning could be roughly categorized into four kinds: generative approaches, data augmentation approaches, memory-based approaches, and meta-learning approaches.

A. Generative Approaches

Most early methods [11]–[13], [28] use generative models to model geometric configurations of objects. For example, Li *et al.* [11] proposed to represent objects by probabilistic models on local features where the parameters are computed on seen categories and generalized to novel categories as prior knowledge. Wong and Yuille [12] proposed to build an AND–OR graph model to represent objects as a composition of meaningful images patches based on the assumption that meaningful patches are shared by seen categories and those novel categories. Reference [13] proposed Bayesian program learning to represent concepts as a generative model, building concepts hierarchically from subparts, parts, and spatial relations. As these approaches represent objects as a composition of parts, a large dictionary of common parts shared by all categories is an essential prerequisite. For unconstrained objects with vast variations, common parts are hard to define. Hence, this kind of methods usually perform well on datasets with limited variations, e.g., handwritten digits, but cannot handle unconstrained objects with vast variations.

B. Data Augmentation Approaches

The key challenge of few-shot learning lies in lacking data and sufficient variations. Thus, a direct solution to few-shot learning is to generate additional data. Traditional skills, e.g., flipping, cropping, and color jittering, can achieve the goal of augmenting data. However, data produced by these ways have high similarity, leading to limited efficacy. Recently, a few works [29]–[33] proposed complex parametric models to generate synthetic data in image domain or in feature domain. For example, Chen *et al.* [32] designed a novel image deformation network, which learns to synthesize additional images by fusing a pair of reference images. Wang *et al.* [33] proposed a GAN-like generator to hallucinate additional training images from a pair of noise and original image. Chen *et al.* [30] proposed a dual TriNet which augments representation in semantic space by introducing slight perturbation and then projects augmented representation into feature space. Despite performance improvement, synthesized data might contain artifacts and miss the necessary details of real data, which is the limitation of these kinds of methods. Besides, how to ensure the diversity of generated data is another big challenge for data augmentation approaches.

C. Memory-Based Approaches

This kind of methods [25], [34]–[37] exploit memory-augmented networks [38]–[40] for few-shot learning. Memory modules in these methods are used to store past experience or knowledge that can be used for novel tasks. For instance, Santoro *et al.* [34] formulated a few-shot classification task as a sequence classification task and used an external memory to hold examples seen in the past and classify new examples. Cai *et al.* [25] adopted a memory module to store the encoded contextual information of a novel task, which is then applied by a contextual learner to generate parameters of the contextual embedding architecture. Xu *et al.* [35] used an external memory to store the information of large-scale machine-labeled datasets and extracted useful related information from this external memory to update another abstraction memory that is retrieved to classify new queries. Memory module makes the network being able to store more useful experience and knowledge, which is helpful for few-shot learning. However, how to internalize past experience into memory and how to utilize the memory are the bottleneck of these methods. In addition, training a memory-augmented network is also a challenging problem at present.

D. Meta-Learning Approaches

Lately, quite a few works [18], [19], [22], [25], [41] follow the paradigm of meta-learning to solve few-shot problems considering its good generalization ability on novel tasks. These works optimize a meta-learner with an abundance of few-shot tasks sampled from the training set, aiming to guide a learner adapting to a novel task with few shots at the testing stage. A quite inspiring work MAML [19] proposes to learn a nice model initialization so that a small number of update steps with few shots can produce good generalization performance on a novel task. To facilitate the training process that involves high-order derivative, the first-order approximation is proposed to avoid computing high-order derivative [20]. Meta-SGD [42] aims to learn an optimizer as a substitute for traditional SGD, which takes as input the given several examples and outputs the learning rate and update the direction of parameters. Some other works [23], [25]–[27], [41], [43], [44] learn a task-agnostic predictor to directly predict model parameter weights instead of fine-tuning them like in MAML [19] and Meta-SGD [42]. For example, Qiao *et al.* [43] learned a category-agnostic predictor, which transforms the activation of few examples to classifier weights. Gidaris and Komodakis [23] proposed to use a denoising autoencoder network to generate classifier weights for novel categories with a few given samples. Li *et al.* [26] introduced a novel module named MetaNet to predict the weights of the feature extractor from few shots to extract task-dependent feature representation.

There are also some works, such as MatchingNetwork [16], PrototypicalNetwork [17], RelationNetwork [18], TADAM [24], and MetaMetric [8], [45], which endeavor to learn a task-agnostic or task-adaptive metric in meta-learning manner under which the nearest neighbor classifier could be built for unseen categories. For instance, RelationNetwork [18] learns to learn a task-agnostic deep distance metric generalized well to novel tasks. MetaMetric [45] learns a task-adaptive metric via training a meta-learner to fine-tune the parameters of a metric-based model. Recent [8] builds up a probabilistic-based classifier by calculating the probabilities of queries sampling from bias-eliminated category-specific distributions estimated via variational inference, which is proven to be an extension of weighted Euclidean distance metric.

The core essence of meta-learning approaches lies in learning to learn and quick adaption to novel data, and this makes it achieve quite promising performance on few-shot tasks.

III. Method

In this section, we first give the problem definition of few-shot classification and then introduce our basic idea followed by the detailed description of our proposed method. Finally, intensive discussions are provided to present the differences of our method from the existing methods.

A. Problem Definition

The goal of few-shot classification is to learn a good classification model from a few labeled examples. It is usually formulated as an N-way K-shot classification task, i.e., recognizing N categories given K samples per category, and K is usually very small, e.g., 1 and 5.

Generally, in a typical meta-learning setting for few-shot classification problems, three types of datasets are needed: a meta-training set, a meta-validation set, and a meta-testing set. The process of training is conducted on a series of episodes. Each episode is an N-way K-shot task sampled from the meta-training set and contains two parts: a support set containing K examples for each of the sampled N categories and a query set containing examples of the N categories to be classified. The objective of training is to minimize classification error over different tasks to learn a meta-learner that learns how to learn a well-performing model from few labeled examples. In the validation or testing phase, a number of N-way K-shot tasks are randomly sampled from the meta-validation or meta-testing set, and the averaged classification accuracy on these tasks is taken as the final performance on few-shot classification.

B. Basic Idea

Our goal is to learn a classifier–predictor to generate classifier weights from few samples. The first step is to model the classifier–predictor, i.e., the relationship between classifier weights and few samples. It is well known that the embeddings of images in the same category spatially cluster around the corresponding classifier weights. Based on the observation that the embeddings and classifier weights for different categories have a similar structure, Qiao *et al.* [43] modeled a category-agnostic generator T parameterized by θ, denoted as T_θ, to transform feature embeddings to the classifier weights given a good feature extractor, which is formulated as follows:

$$\mathbf{w}_n = T_\theta(\mathbf{z}_n), \quad \mathbf{z}_n = E(\mathbf{I}_n) \tag{1}$$

where $\mathbf{I}_n$ is an image of the nth category, $\mathbf{z}_n$ is the normalized feature embedding of $\mathbf{I}_n$ extracted by feature extractor E, and $\mathbf{w}_n$ is the normalized classifier weights corresponding to the nth category. With the classifier weights $\{\mathbf{w}_n | n = 1, \ldots, N\}$, a classifier can be formed as follows:

$$p(y = i|\mathbf{I}) = \frac{e^{\langle \mathbf{z}, \mathbf{w}_i \rangle}}{\sum_{n=1}^{N} e^{\langle \mathbf{z}, \mathbf{w}_n \rangle}}. \tag{2}$$

In [43], the generator T_θ is learned on seen categories in the training set and generalized to novel categories. However, as we all know, different categories have different intraclass variations. Despite good generalization, a category-agnostic generator is certainly a compromise for all categories. This means that a category-agnostic generator can only capture the common variations shared by different categories at a

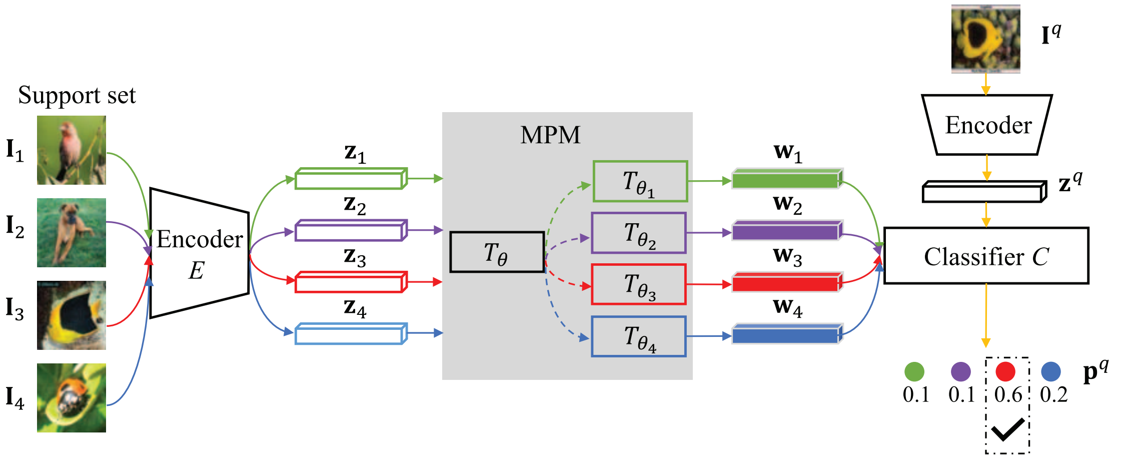


Fig. 1. Overview of our method consisting of three parts: an encoder E to extract informative feature, an MPM to adapt a universal generator T_θ to category-specific ones $\{T_{\theta_n}|n = 1, \ldots, N\}$ building up our task-adaptive classifier–predictor, and a classifier C for final classification. The parameters of classifier C are predicted by those category-specific generators $\{T_{\theta_n}|n = 1, \ldots, N\}$ (best viewed in color).

coarse level, for example, translation and affine transformation. It ignores the discrepancy on intraclass variation between different categories, for example, rigid object Building and nonrigid object Dog. Hence, there should be another category-specific mapping function G_n parameterized by ϕ_n, denoted as G_{ϕ_n}, to capture the individual variation of the nth category at a fine-grained level, which can be formulated as follows:

$$\mathbf{w}_n = G_{\phi_n}(T_\theta(\mathbf{z}_n)) \triangleq T_{\theta_n}(\mathbf{z}_n). \tag{3}$$

With the above idea, we propose to learn category-specific generators, i.e., task-adaptive classifier–predictor, to predict the classifier weights for novel tasks. In a nutshell, a novel MPM is introduced as the meta-learner to learn how to learn category-specific generators with few shots for a given task. To be specific, given an N-way K-shot task $\mathcal{T}$ with a support set $S = \{(\mathbf{I}_n^k, Y_n^k)|n = 1, \ldots, N; k = 1, \ldots, K\}$, the MPM learns to update a universal generator T_θ to multiple category-specific ones $\{T_{\theta_n}|n = 1, \ldots, N\}$ with the given support set under the guide of a newly proposed center-uniqueness loss function, i.e.,

$$\{T_{\theta_1}, \ldots, T_{\theta_N}\} = \text{MPM}\big(T_\theta, \{\mathbf{z}_n^k|n = 1, \ldots, N; k = 1, \ldots, K\}\big) \tag{4}$$

where $\mathbf{I}_n^k$ is the kth sample of the nth category, Y_n^k is the ground-truth label of $\mathbf{I}_n^k$, and $\mathbf{z}_n^k = E(\mathbf{I}_n^k)$ is the normalized embedding of $\mathbf{I}_n^k$. Each generator T_{θ_n} is specialized for an unseen category to generate the corresponding classifier weights $\mathbf{w}_n$ from embeddings of few samples and thus can capture the peculiar characteristics of that novel category. Therefore, the classifier built up by these generated classifier weights $\{\mathbf{w}_n|n = 1, \ldots, N\}$ can perform well on the novel task.

C. Framework

As shown in Fig. 1, the overall network of our method consists of three parts: an encoder E to extract informative feature embeddings, the newly introduced MPM to learn task-adaptive classifier–predictor, and a classifier C for final classification. The parameter weights of the final classifier are predicted by the task-adaptive classifier–predictor learned by MPM. Briefly speaking, our MPM learns a good universal generator T_θ and also learns how to adaptively update it to multiple category-specific ones $\{T_{\theta_n}|n = 1, \ldots, N\}$ with the novel center-uniqueness loss.

What to be learned in the training phase are the encoder E and the universal generator T_θ and how to quickly update the universal generator T_θ to category-specific ones. To optimize them, we employ a two-stage training pipeline, such as [27] and [43]. In the first stage, the encoder E is optimized with a conventional classification objective on the whole meta-training set. In the second stage, the learned encoder E keeps fixed and the newly introduced module MPM is optimized to obtain a good universal generator T_θ, which can be easily updated to well-performing category-specific generators. Similar to other meta-learning methods including MAML [19], Reptile [20], and Meta-SGD [42], we adopt the meta-training strategy, i.e., alternate optimization between inner loop and outer loop, to optimize the MPM module. The optimization in the inner loop and outer loop plays different roles. The optimization in the inner loop learns the learner, i.e., category-specific generators for a given task; The optimization in the outer loop learns the meta-learner, i.e., the universal generator across a variety of sampled tasks. Through iterative inner loop and outer loop optimization, finally, a universal generator T_θ that can be easily updated to category-specific generators will be obtained.

Specifically, at each iteration, a batch of few-shot tasks are sampled from meta-training set. Category-specific generators of each task are first initialized with the universal generator T_θ. Then, in the inner loop, category-specific generators of each task are updated through gradient descent by minimizing the sum of the cross-entropy loss and a novel center-uniqueness loss. In the outer loop, the universal generator T_θ is updated by back-propagating gradients from category-specific generators

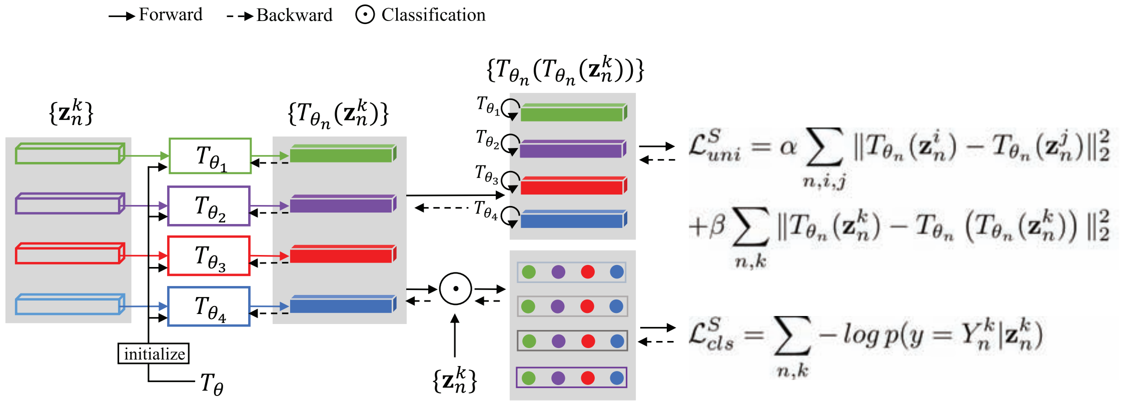

Forward
Backward
Classification
$\{\mathbf{z}_n^k\}$
$\{T_{\theta_n}(\mathbf{z}_n^k)\}$
$\{T_{\theta_n}(T_{\theta_n}(\mathbf{z}_n^k))\}$
T_{θ_1}
T_{θ_2}
T_{θ_3}
T_{θ_4}
initialize
T_θ
$\{\mathbf{z}_n^k\}$
$\mathcal{L}_{uni}^S = \alpha \sum_{n,i,j} \|T_{\theta_n}(\mathbf{z}_n^i) - T_{\theta_n}(\mathbf{z}_n^j)\|_2^2$
$+\beta \sum_{n,k} \|T_{\theta_n}(\mathbf{z}_n^k) - T_{\theta_n}(T_{\theta_n}(\mathbf{z}_n^k))\|_2^2$
$\mathcal{L}_{cls}^S = \sum_{n,k} -\log p(y = Y_n^k|\mathbf{z}_n^k)$

Fig. 2. Illustration of learning category-specific generators of the classifier–predictor in inner loop. All category-specific generators $\{T_{\theta_n}|n = 1, \ldots, N\}$ are all initialized with the given universal generator T_θ and then optimized by minimizing the sum of the classification loss L_{cls}^S and the center-uniqueness loss L_{uni}^S computed on the support set (best viewed in color).

of all sampled tasks to obtain a new universal generator. After multiple iterations over a variety of tasks, the universal generator T_θ would converge to an optimized one that is beneficial for all tasks to get their category-specific generators.

1) Learn Category-Specific Generators in Inner Loop: As analyzed before, category-specific generators can better capture the individual characteristics of each category than a universal one. In this step, our MPM updates category-specific generators $\{T_{\theta_n}|n = 1, \ldots, N\}$ initialized with the universal generator T_θ, as shown in Fig. 2. The universal generator T_θ and the category-specific generator T_{θ_n} are parameterized by θ and θ_n, respectively.

As the classifier weights predicted by category-specific generators $\{T_{\theta_n}|n = 1, \ldots, N\}$ serve as the parameters of the classifier for final classification, it is essential to make sure that the predicted classifier is discriminating. Conventionally, similar to MAML [19], Reptile [20], and LEO [22], the cross-entropy loss $\mathcal{L}_{\text{cls}}^S$ computing on the support set S is used to ensure the discrimination power of predicted classifier weights, which takes the form

$$\mathcal{L}_{\text{cls}}^S(C_\Theta) = \text{avg}\left(\sum_{n=1}^{N}\sum_{k=1}^{K} -\log p\left(y = Y_n^k|\mathbf{I}_n^k; C_\Theta\right)\right) \quad (5)$$

where C_Θ is the classifier predicted by the current category-specific generators parameterized by $\Theta = [\theta_1, \ldots, \theta_N]$ and $p(y = i|\mathbf{I}; C_\Theta)$ stands for the probability that the image $\mathbf{I}$ with embedding $\mathbf{z}$ belongs to the ith category predicted by C_Θ. The probability $p(y = i|\mathbf{I}; C_\Theta)$ is computed from a softmax over cosine similarities between the embedding and predicted classifier weights, i.e.,

$$p(y = i|\mathbf{I}; C_\Theta) = \frac{e^{\langle \mathbf{z}, \mathbf{w}_i\rangle/\tau}}{\sum_{n=1}^{N} e^{\langle \mathbf{z}, \mathbf{w}_n\rangle/\tau}}, \quad \Theta = [\theta_1, \ldots, \theta_N] \quad (6)$$

where τ is the temperature hyperparameter and the classifier weights $\mathbf{w}_n$ is obtained by averaging multiple predictions from multiple shots formulated as

$$\mathbf{w}_n = \frac{1}{K}\sum_{k=1}^{K} \tilde{\mathbf{w}}_n^k, \quad \tilde{\mathbf{w}}_n^k = T_{\theta_n}\left(\mathbf{z}_n^k\right). \quad (7)$$

As only a few examples are given in support set, if we directly minimize the objective function in (5), the category-specific generators $\{T_{\theta_n}|n = 1, \ldots, N\}$ would be easy to overfit the given samples, leading to a biased solution. To get more accurate category-specific generators, we propose a novel center-uniqueness loss function $\mathcal{L}_{\text{uni}}^S$ to regularize the update direction of these category-specific generators.

First, the classifier weights predicted from different samples should be as identical and unique as possible

$$T_{\theta_n}\left(\mathbf{z}_n^i\right) = T_{\theta_n}\left(\mathbf{z}_n^j\right) = \mathbf{w}_n \quad (8)$$

where $\mathbf{z}_n^i$ and $\mathbf{z}_n^j$ are two samples of the nth category. Besides, the classifier weights $\mathbf{w}_n$ is the center of the nth category when all embeddings are normalized to a unit hypersphere as discussed in [46]. Also, the center $\mathbf{w}_n$ can be seen as a prototype sample of the nth category. Therefore, following (8), if this special sample $\mathbf{w}_n$ is fed into the generator T_{θ_n}, the predicted classifier weights should be itself, that is to say, $\mathbf{w}_n$ is a fixed point of the corresponding category-specific generator T_{θ_n}, i.e.,

$$T_{\theta_n}(\mathbf{w}_n) = \mathbf{w}_n. \quad (9)$$

From (8) and (9), we can further derive that

$$T_{\theta_n}\left(\mathbf{z}_n^k\right) = T_{\theta_n}\left(T_{\theta_n}\left(\mathbf{z}_n^k\right)\right) \quad \forall\ k \in \{1, \ldots, K\} \quad \forall\ n \in \{1, \ldots, N\}. \quad (10)$$

Based on (8) and (10), our novel center-uniqueness loss function $\mathcal{L}_{\text{uni}}^S$ is designed as a combination of the two terms $\mathcal{L}_{\text{eq}}^S$ and $\mathcal{L}_{\text{fix}}^S$, i.e.,

$$\mathcal{L}_{\text{uni}}^S(C_\Theta) = \alpha\mathcal{L}_{\text{eq}}^S(C_\Theta) + \beta\mathcal{L}_{\text{fix}}^S(C_\Theta) \quad (11)$$

$$\mathcal{L}_{\text{eq}}^S(C_\Theta) = \text{avg}\left(\sum_{n=1}^{N}\sum_{i=1}^{K}\sum_{j=1}^{K}\left\|T_{\theta_n}\left(\mathbf{z}_n^i\right) - T_{\theta_n}\left(\mathbf{z}_n^j\right)\right\|_2^2\right) \quad (12)$$

$$\mathcal{L}_{\text{fix}}^S(C_\Theta) = \text{avg}\left(\sum_{n=1}^{N}\sum_{k=1}^{K}\left\|T_{\theta_n}\left(\mathbf{z}_n^k\right) - T_{\theta_n}\left(T_{\theta_n}\left(\mathbf{z}_n^k\right)\right)\right\|_2^2\right) \quad (13)$$

where α and β are hyperparameters to balance two terms.

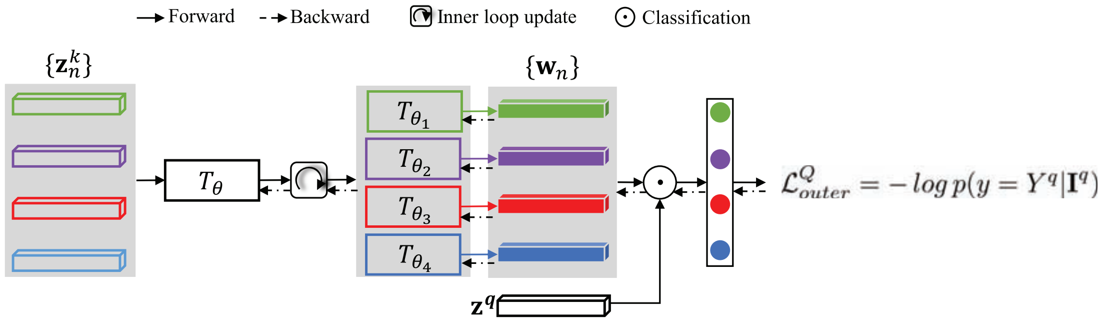


Fig. 3. Illustration of updating the universal generator in outer loop. The universal generator T_θ is updated by minimizing classification error on the query set given the category-specific generators $\{T_{\theta_n}|n = 1, \ldots, N\}$ optimized in the inner loop (best viewed in color).

Eventually, the overall loss function for optimizing the category-specific generators is formulated as the sum of cross-entropy loss $\mathcal{L}_{\text{cls}}^S$ and center-uniqueness loss $\mathcal{L}_{\text{uni}}^S$ as follows:

$$\min_{\Theta=[\theta_1,\ldots,\theta_N]} \mathcal{L}_{\text{inner}}^S(C_\Theta) = \mathcal{L}_{\text{cls}}^S(C_\Theta) + \mathcal{L}_{\text{uni}}^S(C_\Theta) \tag{14}$$

where $\mathcal{L}_{\text{cls}}^S$ aims for discrimination ability and $\mathcal{L}_{\text{uni}}^S$ ensures the robustness and consistency of the classifiers predicted from different few samples. By minimizing the overall loss in (14), the category-specific generators $\{T_{\theta_n}|n = 1, \ldots, N\}$ parameterized by $\{\theta_n|n = 1, \ldots, N\}$ can be obtained.

Specifically, all category-specific generators $\{T_{\theta_n}|n = 1, \ldots, N\}$ are first initialized with the universal one T_θ and then are updated using the gradient descent algorithm by minimizing the overall loss in (14), i.e.,

$$\begin{cases} \theta_n^0 = \theta \\ \theta_n^t = \theta_n^{t-1} - \zeta \nabla_{\theta_n} \mathcal{L}_{\text{inner}}^S(C_\Theta) \end{cases} \tag{15}$$

where θ_n^t represents θ_n at the tth update step. In the case of only one update step, the update process can be simplified as

$$\theta_n = \theta - \zeta \nabla_{\theta_n} \mathcal{L}_{\text{inner}}^S(C_\Theta)|_{\theta_n=\theta}. \tag{16}$$

In this way, parameters $\{\theta_n|n = 1, \ldots, N\}$ of category-specific generators $\{T_{\theta_n}|n = 1, \ldots, N\}$ are updated from the same starting point θ toward their respective optimal solutions. After several steps of gradient descent, optimal category-specific generators in expectation are obtained.

2) Update Universal Generator in Outer Loop: Similar to MAML [19], Reptile [20], and LEO [22], the universal generator T_θ is optimized across many tasks randomly sampled from the meta-training set. To be specific, the universal generator T_θ is randomly initialized at the beginning. At each iteration, i.e., one circle of inner loop and outer loop, a batch of tasks $\mathcal{T}$ are randomly sampled from the meta-training set, each of which consists of a support set S and a query set Q. The universal generator T_θ is first adapted to category-specific ones $\{T_{\theta_n}|n = 1, \ldots, N\}$ for each task based on the support set S in inner loop. As the adapting process is differentiable, the universal generator T_θ is further updated via SGD by minimizing the classification error of learned category-specific generators $\{T_{\theta_n}|n = 1, \ldots, N\}$ of all sampled tasks on their query set Q. The outer loop aims for an updated universal generator T_θ that is more beneficial for learning category-specific generators. The objective of updating the universal generator T_θ is formulated as follows:

$$\begin{aligned} \min\nolimits_\theta & \sum_{\mathcal{T}\sim p(\mathcal{T})} \mathcal{L}_{\text{outer}}^Q(C_\Theta) \\ &= \sum_{\mathcal{T}\sim p(\mathcal{T})} \mathcal{L}_{\text{outer}}^Q\left(C_{\left[\theta-\zeta\nabla_{\theta_n}\mathcal{L}_{\text{inner}}^S(C_\Theta)\big|_{\theta_n=\theta}\right]}\right) \\ &= \sum_{\mathcal{T}\sim p(\mathcal{T})} \frac{1}{M} \sum_{m=1}^{M} -\log p\left(y = Y_m^q|\mathbf{I}_m^q; C_\Theta\right) \end{aligned} \tag{17}$$

where Y_m^q is the ground-truth label of $\mathbf{I}_m^q$ in query set Q and C_Θ is the classifier with the classifier weights predicted by the learned category-specific generators $\{T_{\theta_n}|n = 1, \ldots, N\}$ as parameters. An illustration of updating T_θ in outer loop is shown in Fig. 3.

Overall, by optimizing category-specific generators in inner loop and updating the universal generator in outer loop across different tasks, we can obtain our MPM that not only learns a good universal generator T_θ for easy adaptation but also learns how to adaptively update this universal generator to multiple category-specific ones $\{T_{\theta_n}|n = 1, \ldots, N\}$ quickly and accurately for a novel task with the newly proposed center-uniqueness loss. The algorithm of training our MPM is summarized in Algorithm **1**.

Given a novel task in the testing phase, as described in Algorithm **2**, the embeddings of few samples in support set are first extracted by the encoder E. With the embeddings of samples in support set, our MPM updates the learned universal generator to category-specific ones in few steps. Each of these generators is specialized for an unseen category in the task and is used to predict the corresponding classifier weights. Then, the classifier for this novel task is constructed with the predicted classifier weights as parameters to recognize new queries.

D. Discussion

Our proposed approach lies in the line of exploiting meta-learning to learn to predict target model weights. Compared with other methods lying in this line [23], [24], [26], [43], the classifier–predictor of our method is task-adaptive,

Algorithm 1 Training Algorithm of MPM

```
Require: $p(\mathcal{T})$: distribution over tasks
Require: $U, \zeta, \eta, \alpha, \beta$: hyperparameters
Require: $E$: trained encoder which keeps fixed
randomly initialize parameters $\theta$
while not done do
    Sample batch of few shot tasks $\mathcal{T} \sim p(\mathcal{T})$
    for each task $\mathcal{T}$ do
        Initialize $\{\theta_n\}_{n=1}^{N}$ with $\theta$, $\Theta = [\theta_1, \ldots, \theta_N]$
        for $0 < t < U$ do                                  ▷ Inner loop
            Compute $\mathcal{L}_{inner}^{S}(C_\Theta)$ using Eq. (14)
            Update $\theta_n^t = \theta_n^{t-1} - \zeta \nabla_{\theta_n} \mathcal{L}_{inner}^{S}(C_\Theta)$
        end for
        Compute $\mathcal{L}_{outer}^{Q}(C_\Theta)$ using Eq. (17)
    end for
    Update $\theta = \theta - \eta \nabla_\theta \sum_{\mathcal{T}} \mathcal{L}_{outer}^{Q}(C_\Theta)$      ▷ Outer loop
end while
```

Algorithm 2 Testing Algorithm

```
Require: $E, T_\theta$: trained encoder and universal generator
Require: $U, \zeta, \alpha, \beta$: hyperparameters
Require: $\mathcal{T}$: a task with support set $S$ and a query $q$
Extract feature embeddings using encoder $E$
Initialize $\{\theta_n\}_{n=1}^{N}$ with $\theta$, $\Theta = [\theta_1, \ldots, \theta_N]$
for $0 < t < U$ do                                          ▷ Finetune
    Compute $\mathcal{L}_{inner}^{S}(C_\Theta)$ using Eq. (14)
    Update $\theta_n^t = \theta_n^{t-1} - \zeta \nabla_{\theta_n} \mathcal{L}_{inner}^{S}(C_\Theta)$
end for
Compute classifier weights using Eq. (7)
Classify $q$ using Eq. (6)
```

i.e., determined adaptively by tasks. The most related works to ours are MAML [19], [43]. Learning a universal generator and then adaptively updating it to category-specialized ones in our method can also be seen as learning an initialization and fine-tuning further for novel tasks, such as MAML [19]. However, MAML [19] directly fine-tunes the model parameters for novel tasks. Differently, our MPM optimizes and fine-tunes the predictor of classifier weights, which has higher similarity between different tasks [43] and thus is easier to adapt to novel tasks than MAML [19]. As for [43] which also learns to predict classifier weights from few samples, its classifier–predictor is a universal generator shared by all unseen categories in novel tasks. This implicitly assumes that all categories have the same intraclass variation, which apparently hardly holds especially for those vastly different categories, such as Building and Dog. By contrast, our MPM, able to adaptively learn specialized generators for each unseen category in novel tasks, is naturally a better way to predict an accurate classifier.

On the other hand, our method can also be seen as learning task-adaptive metric as the classification of a query sample is determined by its cosine similarity with predicted classifier weights. Compared with existing methods [16]–[18] which learn a task-agnostic metric, our method able to learn specialized metrics for novel tasks is superior for few-shot classification. Compared with [45] which also learns task-adaptive metric by fine-tuning the parameters of a metric-based model, our method fine-tunes the classifier–predictor, which has higher similarity between different tasks and thus is more easier to adapt to novel tasks. In addition, our method is applicable for both one-shot and multiple-shot scenario, while [45] can only work in multiple-shot scenarios.

IV. Experiments

We evaluate our method on two commonly used benchmarks for few-shot classification, i.e., miniImageNet and tieredImageNet. Details about datasets and experimental settings are introduced next, followed by a comprehensive ablation study and comparison with the state-of-the-art methods. Finally, we give some visualization results for a better understanding of our method.

A. Datasets

*1) mini*ImageNet:* This dataset, originally proposed in [16], is a subset randomly sampled from ILSVRC12 [47]. It contains 100 categories with 600 images per category. To compare with other works fairly, we follow the splits proposed in [21], which splits all 100 categories into three subsets. One split with 64 categories is used as meta-training set for training. The other two splits are used as meta-validation set and meta-testing set containing 16 and 20 categories, respectively.

*2) tiered*ImageNet:* This dataset is proposed in [48] for few-shot classification. It is a larger subset with 608 categories of ILSVRC12 [47]. All 608 categories are grouped into 34 broader categories corresponding to 34 high-level nodes in the ImageNet hierarchy [49]. Each broader category contains about 10–30 categories with around 1300 images per category. It is also split into three subsets: a meta-training set, a meta-validation set, and a meta-testing set with 20, 6, and 8 broader categories, respectively. Categories in three splits are totally different making this dataset more challenging.

B. Experimental Setting

1) Model Configuration: For fair comparison, following existing works [17], [22], [27], [43], we select three architectures for the encoder E: a shallow convolutional network denoted as ConvNet4 and two larger networks, namely, *ResNet12* [50] and *WRN-28-10* [51]. The details about these three architectures are described as follows.

ConvNet4 is made up of four convolutional blocks, each of which contains four layers with the order of CONV-BN-RELU-MaxPool. All convolutional layers have 64 filters. Following [27], the RELU layer in the last block is removed for better performance. Images are resized to the resolution of 84×84 and the output of the last block is flattened into 1600-D representation. ResNet12 contains four residual blocks, each of which is composed of three convolutional building blocks and a shortcut connection. Each convolutional building block comprises three layers with the order of BN-RELU-CONV. Each residual block is followed by a max-pooling layer. The number of filters of convolutional layers in four residual

TABLE I

ABLATION STUDY. COMPARISON OF BASELINE METHODS WITH CONVNET4 AS ENCODER. ALL ACCURACY RESULTS ARE AVERAGED OVER 600 TEST EPISODES AND REPORTED WITH 95% CONFIDENCE INTERVALS

Method	Type	*mini*ImageNet		*tiered*ImageNet	
		1-shot	5-shot	1-shot	5-shot
MAML [19]	Finetune classifier	48.70%	63.11%	51.67%	70.30%
[43]	Task-agnostic classifier-predictor	54.54%	67.87%	56.94%	73.32%
[43]+L_{eq}+L_{fix}	Task-agnostic classifier-predictor	56.69%	72.98%	58.36%	75.19%
MPM-S+L_{cls}+L_{eq}+L_{fix}	Task-adaptive classifier-predictor	55.84%	73.11%	57.48%	74.97%
MPM+L_{cls}	Task-adaptive classifier-predictor	56.76%	73.05%	59.11%	75.52%
MPM+L_{cls}+L_{eq}		57.35%	73.83%	**59.61%**	76.19%
MPM+L_{cls}+L_{fix}		57.25%	73.30%	58.70%	75.90%
MPM+L_{cls}+L_{eq}+L_{fix} (ours)		**57.59%**	**74.02%**	59.44%	**76.59%**

blocks is 64, 128, 256, and 512, respectively. Considering larger capacity of ResNet12, images are resized to a higher resolution of 112 $\times 112$ and encoded as 512-D representation. WRN-28-10 is a wider and deeper network with 28 convolutional layers. For details of WRN-28-10, please refer to [51]. With WRN-28-10 as the encoder, input images are resized to 112×112 and encoded as 640-D representation.

In all experiments, all generators, including T_θ and $\{T_{\theta_n} | n = 1, \ldots, N\}$, are modeled as one fully connected layer for simplicity.

2) Training Details: As said before, we employ a two-stage training pipeline as in [27], i.e., training the encoder in the first stage and then training MPM with the encoder fixed in the second stage. When training the encoder in the first stage, dropout with $p_{\text{keep}} = 0.4$ is used inside each residual block, especially for WRN-28-10. In the second stage of training MPM, the classification error of queries in the outer loop is computed over all categories in the meta-training set, i.e., 64 categories for miniImageNet, instead of sampled N novel categories. The corresponding classifier weights of novel categories are predicted and the remaining classifier weights are from the first stage. This way makes the classifier weights predicted by category-specific generators more accurate, leading to better performance. In one-shot scenario, as there is only one sample per category in support set, the first term $\mathcal{L}_{\text{eq}}$ of center-uniqueness loss $\mathcal{L}_{\text{uni}}$ becomes constant zero and ineffective in learning category-specific generators. To alleviate this problem, we flip the samples in support set horizontally so that each category has two samples making the term $\mathcal{L}_{\text{eq}}$ take effect.

In the whole training phase, we only use the meta-training set as training data and the meta-validation set is used to tune hyperparameters. Two skills, i.e., random crop and random horizontal flip, are used to augment training data. For training our model, we use an SGD optimizer with momentum 0.9 and weight decay 0.0005. The initial learning rate η for outer loop optimization is set to 0.1 and decayed by 0.1 for every 15 000 iterations. The learning rate ζ for inner loop optimization keeps 1.0 and the number of update steps U in the inner loop is set to 2. The hyperparameter α is empirically set to 1.0 in all experiments. Also, the hyperparameter β is set to 2.5 and 5 for miniImageNet and tieredImageNet, respectively.

As both the outer loop and inner loop involve optimization, updating parameters of T_θ needs to compute high-order gradients. To alleviate computing complexity, we use first-order approximation [19] instead of high-order gradients in our implementation. All experiments are performed using PyTorch.

3) Testing Details: For a fair comparison, we follow the same evaluation setting used by other methods. To be specific, 600 tasks are randomly sampled from the meta-testing set, with each task containing 15 queries per category in both five-way one-shot and five-way five-shot scenarios. Averaged top-1 classification accuracy over these 600 tasks represents the final performance of methods on few-shot classification.

C. Ablation Study

We perform ablation study on both two datasets to assess the effectiveness of our method, including the effect of our MPM used to learn category-specific generators and each term L_{cls}, L_{eq}, L_{fix} in the overall loss function L_{inner}. Here, we use two methods as our baseline: MAML [19], and [43]. MAML [19] learns a good model initialization and fine-tunes model parameters for novel tasks. Method [43] trains a universal generator shared by all categories, i.e., a task-agnostic classifier–predictor, to predict classifier weights. Comparison results are shown in Table I. All the results in Table I are obtained using ConvNet4 as the encoder.

As can be seen, on miniImageNet, the method [43] performs better than MAML [19] by learning a universal generator to generate classifier weights. On tieredImageNet, the improvement of [43] over MAML [19] is slightly lower than on miniImageNet, as the differences between the categories in training and testing sets of tieredImageNet are larger and a universal generator learned from training categories cannot promise accurate classification on testing categories. Besides our method, we also evaluate a variant of our method named MPM-S, which learns a task-adaptive but category-agnostic classifier–predictor through updating a universal generator T_θ to one single adaptive generator instead of multiple category-adaptive generators in the inner loop. As can be seen, both MPM-S and MPM outperform [43] demonstrating the effectiveness of task-adaptive classifier–predictor. In addition, the performance of MPM is higher than MPM-S demonstrating the superiority of category-specific generators.

To investigate the effectiveness of our proposed loss terms L_{eq} and L_{fix}, we evaluate our MPM with different loss terms. As can be seen from Table I, compared with "MPM + L_{cls}" only using cross-entropy loss L_{cls}, the performance of our MPM can be improved by combining either one of

TABLE II

INVESTIGATION OF CORRESPONDENCE BETWEEN GENERATORS AND CATEGORIES. ALL THE RESULTS ARE OBTAINED WITH CONVNET4 AS ENCODER

Method	*mini*ImageNet		*tiered*ImageNet	
	1-shot	5-shot	1-shot	5-shot
RandomOne	51.59%	47.23%	40.26%	44.77%
Ensemble of [43]	55.65%	71.99%	57.97%	74.92%
Ours	**57.59%**	**74.02%**	**59.44%**	**76.59%**

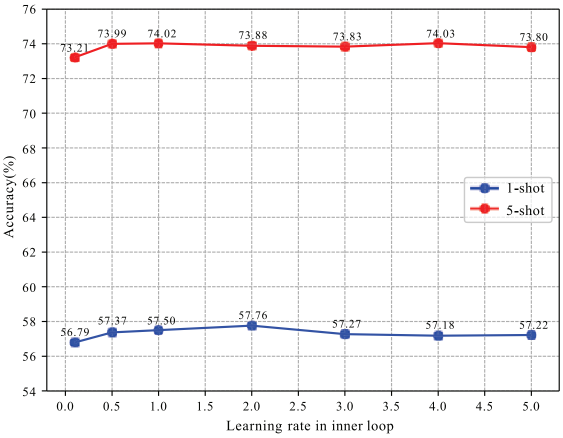


Fig. 4. Performance of our MPM with respect to different learning rates in inner loop.

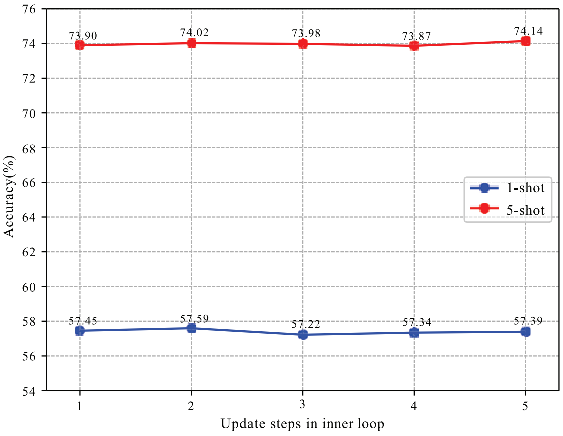


Fig. 5. Performance of our MPM with respect to different update steps in inner loop.

L_{eq} and L_{fix} with L_{cls} (see "MPM + L_{cls} + L_{eq}" and "MPM + L_{cls} + L_{fix}"). As expected, our method "MPM + L_{cls} + L_{eq} + L_{fix}" performs best with all three terms, demonstrating the effectiveness of our newly proposed loss terms L_{eq} and L_{fix}. Furthermore, with these two loss terms to train the task-agnostic classifier–predictor mentioned in [43], the method [43] is also improved by a large margin, verifying the effectiveness and generalization ability of the proposed loss terms L_{eq} and L_{fix} again.

As our method learns category-specific generators, there will be N generators for an N-way task in the testing phase. Despite multiple generators, our method is totally different from ensemble strategy because only one generator is used for each category in our method instead of an ensemble of multiple ones. Nevertheless, we also conduct a comparison with ensemble of multiple category-agnostic generators from [43]. As can be seen from Table II, our method still outperforms "Ensemble of [43]." To further validate the correspondence between categories and generators of our method learned, we randomly pick one generator for each category from N learned category-specific ones denoted as "RandomOne." Significant performance drop of "RandomOne" demonstrates that each category-specific generator of our method learned well match each novel category in a given novel task.

D. Evaluation of Hyperparameters

To investigate the impact of hyperparameters, we evaluate the performance of our method with respect to different choices of important hyperparameters, including the learning rate ζ in the inner loop and the update steps U in the inner loop. The experiments are conducted on the miniImageNet dataset with ConvNet4 as the encoder. Experimental results are shown in Figs. 4 and 5. As can be seen, the performance of our method keeps fairly stable with different choices of hyperparameters within a wide range (learning rate: [0.5, 5.0] and update steps: [1, 5]). This also demonstrates that the improvement of our method mainly comes from our new MPM module and center-uniqueness loss, verifying their effectiveness again.

E. Comparison With State-of-the-Art Works

We compare our method with state-of-the-art works, including optimization-based meta-learning methods such as MAML [19], LEO [22], and recent MetaTransfer [7], metric-based meta-learning methods such as PrototypicalNetwork [17], TADAM [24], and recent [8], and predictor-based meta-learning methods such as [23], [27], and [43]. Among them, [43] with a universal generator is mostly related to ours and can be considered as a baseline of our method.

1) Performance on mini*ImageNet:* From Table III, we can see that with ConvNet4 as the encoder, our method brings a significant improvement over baseline [43], i.e., 3% improvement in one-shot scenario and 6% improvement in five-shot scenario. Also, our method outperforms all methods that use the same architecture except for [52]. Although our method underperforms [52] slightly in a one-shot scenario, it gets more than 3% improvement in the five-shot scenario, mainly because more examples make the center-uniqueness loss work better leading to a more accurate classifier. Besides, with ConvNet4 as the encoder, our method even outperforms several methods [27], [53] with deeper ResNet12 as the encoder in both one- and five-shot scenarios. Moreover, with deeper ResNet12 as the encoder, our method achieves the state-of-the-art performance in both scenarios, illustrating the superiority of our MPM.

2) Performance on tiered*ImageNet:* Experimental results on tieredImageNet are shown in Table IV. As can be seen, with ConvNet4 as the encoder, our method brings about 3% improvement over our baseline method [43]. Besides,

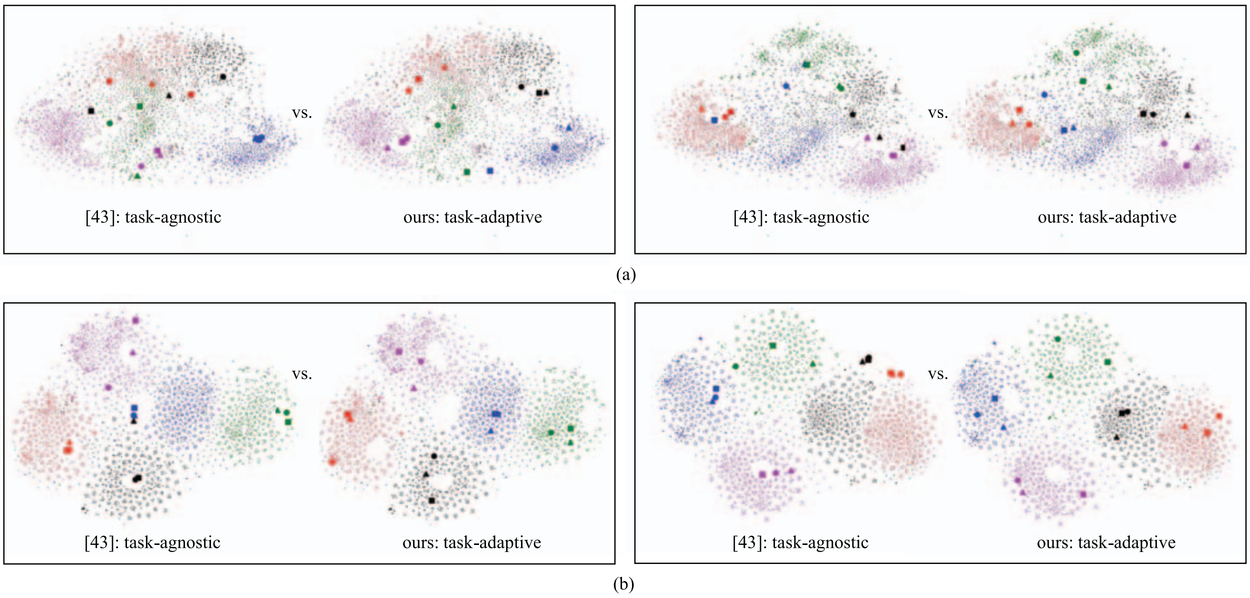


Fig. 6. t-SNE visualization of classifier weights predicted by task-agnostic classifier–predictor [43] and our task-adaptive classifier–predictor. Each black box shows three randomly sampled tasks that involve the same five categories with different support sets. Classifier weights of three tasks are visualized in triangles, circles, and squares, respectively. Different categories are shown in different colors. Points represent feature embeddings (best viewed in color). (a) Five-way one-shot scenario. (b) Five-way five-shot scenario.

TABLE III

Comparison With the State of the Arts on miniImageNet. All Accuracy Results Are Averaged Over 600 Test Episodes. "–" Means Not Reported. "_" Means Baseline and Bold Means Best. †: Using Also the Validation Set for Training

Method	Encoder	1-shot	5-shot
Fully-supervised (Upper bound)	ConvNet4	87.01%	87.01%
MatchingNetwork [16]		43.56%	55.31%
Meta-LSTM [21]		43.44%	60.60%
MAML [19]		48.70%	63.11%
Meta-SGD [42]		50.50%	64.00%
PrototypicalNetwork [17]		49.42%	68.20%
RelationNetwork [18]		50.44%	65.32%
GraphNetwork [54]		50.33%	66.41%
[43]		54.54%	67.87%
PSN [55]		-	66.62%
[52]		**59.00%**	70.90%
MPM(Ours)		57.59%	**74.02%**
SNAIL [53]	ResNet12	55.71%	68.88%
[27]		56.20%	73.00%
TADAM [24]		58.50%	76.70%
MetaTransfer [7]		61.20%	75.50%
[8]		61.23%	77.69%
MPM(Ours)		**62.19%**	**78.51%**
[43]	WRN-28-10	59.60%	73.74%
LEO [22]†		61.76%	77.59%
[23]		61.07%	76.75%
MPM(Ours)		**61.77%**	**78.03%**

TABLE IV

Comparison With State of the Arts on tieredImageNet. All Accuracy Results Are Averaged Over 600 Test Episodes. "–" Means Not Reported. "_" Means Baseline and Bold Means Best. †: Using Also the Validation Set for Training

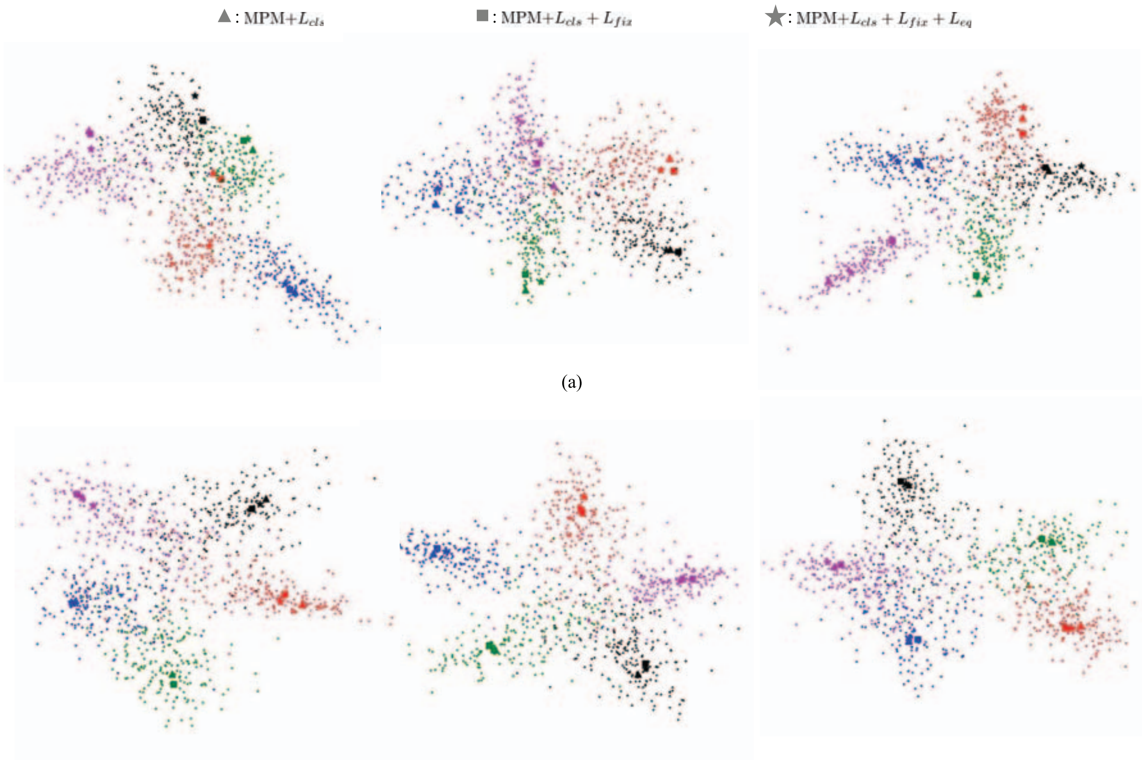
Method	Encoder	1-shot	5-shot
Fully-supervised (Upper bound)	ConvNet4	87.01%	87.01%
MAML [19]		51.67%	70.30%
PrototypicalNetwork [17]		53.31%	72.69%
RelationNetwork [18]		54.48%	71.32%
PSN [55]		-	71.13%
[43]		56.94%	73.32%
MPM(Ours)		**59.44%**	**76.59%**
LEO [22]†	WRN-28-10	66.33%	81.44%
[42](reported by [22])		62.95%	79.34%
[23]		**68.18%**	83.09%
MPM(Ours)		67.58%	**83.93%**

our method outperforms [23], [43] which both learn a task-agnostic classifier–predictor, verifying the necessity and effectiveness of learning task-adaptive classifier–predictor. In addition, the performance of our method is superior to other methods with the same architecture as well. Furthermore, on this more challenging dataset, our method with WRN-28-10 as the encoder achieves the state-of-the-art performance demonstrating the superiority of our method again.

F. Visualization

We visualize the classifier weights predicted by the task-agnostic classifier–predictor mentioned in [43] and by our proposed task-adaptive classifier–predictor in Fig. 6. Each black box shows three randomly sampled tasks that involve the same five categories with different support sets. The classifier weights of three tasks are visualized in different shapes. As can be seen, classifier weights predicted by our task-adaptive classifier–predictor are closer to the center of categories in both one- and five-shot scenarios, i.e., more accurate than the task-agnostic classifier–predictor [43], demonstrating the superiority of our task-adaptive classifier–predictor.

To better understand the effect of our proposed loss terms L_{eq} and L_{fix}, we also visualize the classifier weights predicted by our MPM with different loss functions in Fig. 7. As can be seen, the classifier weights predicted by our MPM with the



(a)

(b)

Fig. 7. Visualization of classifier weights predicted by our MPM with different loss items. Three randomly sampled tasks are illustrated in both five-way one-shot scenario and five-way five-shot scenario. Classifier weights predicted by “MPM + L_{cls},” “MPM + L_{cls} + L_{fix},” and “MPM + L_{cls} + L_{fix} + L_{eq},” are, respectively, visualized with triangles, squares, and stars. Different categories are shown in different colors. Points represent feature embeddings (best viewed in color). (a) Five-way one-shot scenario. (b) Five-way five-shot scenario.

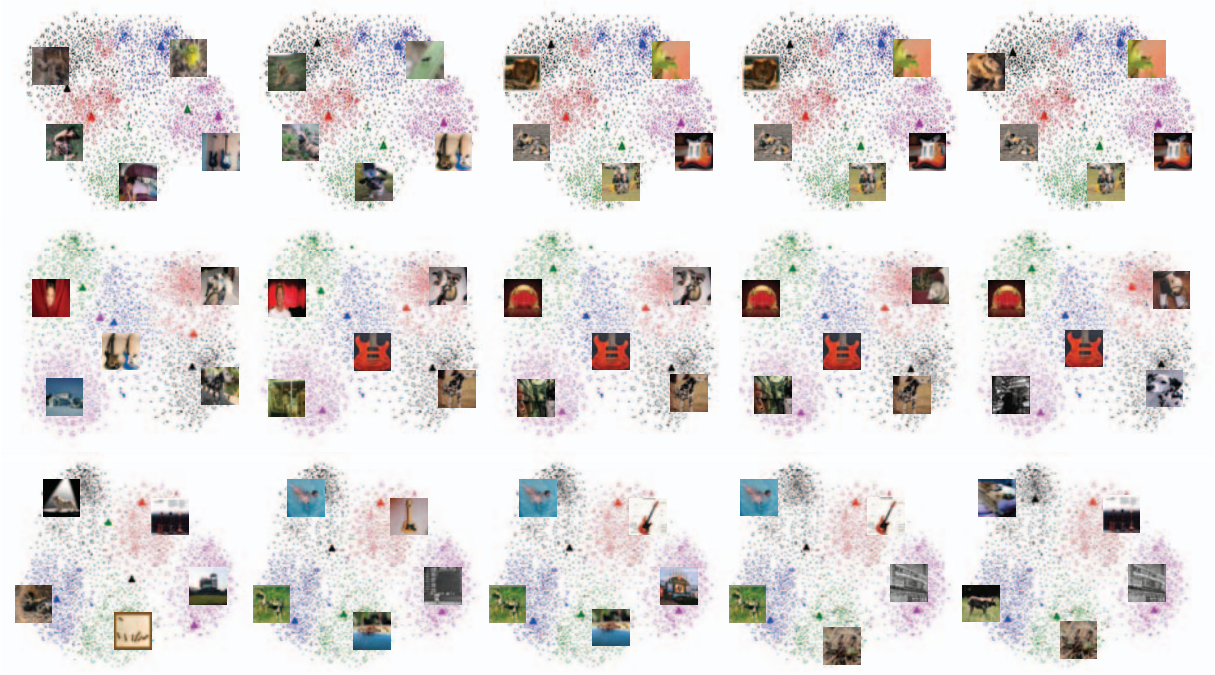


Fig. 8. t-SNE visualization of classifier weights predicted by our method in the five-way K-shot scenario, $K = 1, 2, 3, 4, 5$. Five tasks in each row involve the same five randomly sampled categories with different shots. Different categories are shown in different colors. Points and triangles represent feature embeddings and classifier weights, respectively. Images floating over points are the samples nearest to the predicted classifier weights (best viewed in color).

proposed loss terms are getting more accurate, demonstrating the positive effect of these two loss terms in regularizing the update direction of category-specific generators.

Fig. 8 shows the classifier weights predicted by our method in one-, two-, three-, four-, and five-shot scenarios. Each row shows five randomly sampled tasks that involve the same five categories with different shots. We select images whose embeddings locate nearest to the classifier weights and draw them over the points for better understanding. Expectedly, the predicted classifier weights become more and more accurate with more samples provided in the support set.

V. Conclusion

In conclusion, we propose a novel meta-learning algorithm to learn task-adaptive classifier–predictor for few-shot classification. Our task-adaptive classifier–predictor is composed of multiple category-specific generators, each of which is specialized for one unseen category in a novel task and generates the corresponding classifier weights from few samples of that category. To learn category-specific generators, a novel MPM is introduced to learn how to update a universal generator to category-specific ones under the guide of a newly proposed center-uniqueness loss function. Each category-specific generator in the classifier–predictor is specialized for one unseen category and thus can better capture individual characteristics of each category and predict an accurate classifier. The state-of-the-art performance of our method on both miniImageNet and tieredImageNet verifies the effectiveness of our method and learning task-adaptive classifier–predictor.

VI. Future Work

In this work, our framework can predict the classifier weights given only one shot as input and directly merges the multiple predictions through average operation when multiple shots are given. Later, we will explore to mine the relationship between multiple shots to design better alternatives for merging the predictions from multiple shots. Besides, in this work, category-specific generators are updated independently under the guide of the overall loss function. In the future, we will explore to take the relationships between categories into consideration for further improvement.

References

[1] A. Krizhevsky, I. Sutskever, and G. E. Hinton, “Imagenet classification with deep convolutional neural networks,” in *Proc. Adv. Neural Inf. Process. Syst. (NeurIPS)*, 2012, pp. 1106–1114.

[2] S. Ren, K. He, R. B. Girshick, and J. Sun, “Faster R-CNN: Towards real-time object detection with region proposal networks,” in *Proc. Adv. Neural Inf. Process. Syst. (NeurIPS)*, 2015, pp. 91–99.

[3] K. He, G. Gkioxari, P. Dollár, and R. B. Girshick, “Mask R-CNN,” in *IEEE Int. Conf. Comput. Vis. (ICCV)*, vol. 2017, pp. 2980–2988.

[4] Y. Xian, S. Sharma, B. Schiele, and Z. Akata, “F-VAEGAN-d2: A feature generating framework for any-shot learning,” in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2019, pp. 10275–10284.

[5] Z. Ji, Y. Sun, Y. Yu, Y. Pang, and J. Han, “Attribute-guided network for cross-modal zero-shot hashing,” *IEEE Trans. Neural Netw. Learn. Syst.*, vol. 31, no. 1, pp. 321–330, Jan. 2020.

[6] Y. Yu, Z. Ji, J. Guo, and Z. Zhang, “Zero-shot learning via latent space encoding,” *IEEE Trans. Cybern.*, vol. 49, no. 10, pp. 3755–3766, Oct. 2019.

[7] 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.

[8] J. Zhang, C. Zhao, B. Ni, M. Xu, and X. Yang, “Variational few-shot learning,” in *Proc. IEEE/CVF Int. Conf. Comput. Vis. (ICCV)*, Oct. 2019, pp. 1685–1694.

[9] M. Rostami, S. Kolouri, and P. P. J. McClelland, “Generative continual concept learning,” in *Proc. AAAI Conf. Artificial Intell. (AAAI)*, 2020, pp. 5545–5552.

[10] M. Zhai, L. Chen, F. Tung, J. He, M. Nawhal, and G. Mori, “Lifelong GAN: Continual learning for conditional image generation,” in *Proc. IEEE/CVF Int. Conf. Comput. Vis. (ICCV)*, Oct. 2019, pp. 2759–2768.

[11] 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.

[12] A. Wong and A. Yuille, “One shot learning via compositions of meaningful patches,” in *Proc. IEEE Int. Conf. Comput. Vis. (ICCV)*, Dec. 2015, pp. 1197–1205.

[13] B. M. Lake, R. Salakhutdinov, and J. B. Tenenbaum, “Human-level concept learning through probabilistic program induction,” *Science*, vol. 350, no. 6266, pp. 1332–1338, Dec. 2015.

[14] J. Yosinski, J. Clune, Y. Bengio, and H. Lipson, “How transferable are features in deep neural networks,” in *Proc. Adv. Neural Inf. Process. Syst. (NeurIPS)*, 2014, pp. 3320–3328.

[15] M. D. Zeiler and R. Fergus, “Visualizing and understanding convolutional networks,” in *Proc. Eur. Conf. Comput. Vis. (ECCV)*, 2014, pp. 818–833.

[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. (NeurIPS)*, 2016, pp. 3630–3638.

[17] J. Snell, K. Swersky, and R. S. Zemel, “Prototypical networks for few-shot learning,” in *Proc. Adv. Neural Inf. Process. Syst. (NeurIPS)*, 2017, pp. 4080–4090.

[18] 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.

[19] C. Finn, P. Abbeel, and S. Levine, “Model-agnostic meta-learning for fast adaptation of deep networks,” in *Proc. Int. Conf. Mach. Learn. (ICML)*, 2017, pp. 1126–1135.

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

[21] S. Ravi and H. Larochelle, “Optimization as a model for few-shot learning,” in *Proc. Int. Conf. Learn. Represent. (ICLR)*, 2017.

[22] A. A. Rusu *et al.*, “Meta-learning with latent embedding optimization,” in *Proc. Int. Conf. Learn. Represent. (ICLR)*, 2019, pp. 1–16.

[23] 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.

[24] B. N. Oreshkin, P. R. López, and A. Lacoste, “TADAM: Task dependent adaptive metric for improved few-shot learning,” in *Proc. Adv. Neural Inf. Process. Syst. (NeurIPS)*, 2018, pp. 719–729.

[25] Q. Cai, Y. Pan, T. Yao, C. Yan, and T. Mei, “Memory matching networks for one-shot image recognition,” in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit.*, Jun. 2018, pp. 4080–4088.

[26] H. Li, W. Dong, X. Mei, C. Ma, F. Huang, and B. Hu, “LGM-Net: Learning to generate matching networks for few-shot learning,” in *Proc. Int. Conf. Mach. Learn. (ICML)*, 2019, pp. 3825–3834.

[27] 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.

[28] B. M. Lake, R. Salakhutdinov, and J. B. Tenenbaum, “One-shot learning by inverting a compositional causal process,” in *Proc. Adv. Neural Inf. Process. Syst. (NeurIPS)*, 2013, pp. 2526–2534.

[29] B. Hariharan and R. Girshick, “Low-shot visual recognition by shrinking and hallucinating features,” in *Proc. IEEE Int. Conf. Comput. Vis. (ICCV)*, Oct. 2017, pp. 3037–3046.

[30] Z. Chen, Y. Fu, Y. Zhang, Y.-G. Jiang, X. Xue, and L. Sigal, “Semantic feature augmentation in few-shot learning,” 2018, *arXiv:1804.05298v1*. [Online]. Available: http://arxiv.org/abs/1804.05298v1

[31] E. Schwartz *et al.*, “Delta-encoder: An effective sample synthesis method for few-shot object recognition,” in *Proc. Adv. Neural Inf. Process. Syst. (NeurIPS)*, 2018, pp. 2850–2860.

[32] Z. Chen, Y. Fu, Y.-X. Wang, L. Ma, W. Liu, and M. Hebert, “Image deformation meta-networks for one-shot learning,” in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2019, pp. 8680–8689.

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

[35] Z. Xu, L. Zhu, and Y. Yang, "Few-shot object recognition from machine-labeled web images," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jul. 2017, pp. 5358–5366.

[36] T. Ramalho and M. Garnelo, "Adaptive posterior learning: Few-shot learning with a surprise-based memory module," in *Proc. Int. Conf. Learn. Represent. (ICLR)*, 2019, pp. 1–14.

[37] L. Kaiser, O. Nachum, A. Roy, and S. Bengio, "Learning to remember rare events," in *Proc. Int. Conf. Learn. Represent. (ICLR)*, 2017, pp. 1–10.

[38] J. Weston, S. Chopra, and A. Bordes, "Memory networks," in *Proc. Int. Conf. Learn. Represent. (ICLR)*, 2015, pp. 1–31.

[39] S. Sukhbaatar, A. Szlam, J. Weston, and R. Fergus, "End-to-end memory networks," in *Proc. Adv. Neural Inf. Process. Syst. (NeurIPS)*, 2015, pp. 2440–2448.

[40] A. Miller, A. Fisch, J. Dodge, A.-H. Karimi, A. Bordes, and J. Weston, "Key-value memory networks for directly reading documents," in *Proc. Conf. Empirical Methods Natural Lang. Process.*, 2016, pp. 1400–1409.

[41] X. Wang, F. Yu, R. Wang, T. Darrell, and J. E. Gonzalez, "TAFE-Net: Task-aware feature embeddings for low shot learning," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2019, pp. 1831–1840.

[42] Z. Li, F. Zhou, F. Chen, and H. Li, "Meta-SGD: Learning to learn quickly for few shot learning," 2017, *arXiv:1707.09835*. [Online]. Available: https://arxiv.org/abs/1707.09835

[43] 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.

[44] L. Bertinetto, J. F. Henriques, J. Valmadre, P. H. S. Torr, and A. Vedaldi, "Learning feed-forward one-shot learners," in *Proc. Adv. Neural Inf. Process. Syst. (NeurIPS)*, 2016, pp. 523–531.

[45] D. Wang, Y. Cheng, M. Yu, X. Guo, and T. Zhang, "A hybrid approach with optimization-based and metric-based meta-learner for few-shot learning," *Neurocomputing*, vol. 349, pp. 202–211, Jul. 2019.

[46] H. Wang *et al.*, "CosFace: Large margin cosine loss for deep face recognition," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit.*, Jun. 2018, pp. 5265–5274.

[47] O. Russakovsky *et al.*, "ImageNet large scale visual recognition challenge," *Int. J. Comput. Vis.*, vol. 115, no. 3, pp. 211–252, Dec. 2015.

[48] M. Ren *et al.*, "Meta-learning for semi-supervised few-shot classification," in *Proc. Int. Conf. Learn. Represent. (ICLR)*, 2018, pp. 1–5.

[49] J. Deng, W. Dong, R. Socher, L.-J. Li, K. Li, and L. Fei-Fei, "ImageNet: A large-scale hierarchical image database," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, Jun. 2009, pp. 248–255.

[50] 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.

[51] S. Zagoruyko and N. Komodakis, "Wide residual networks," in *Proc. Brit. Mach. Vis. Conf.*, 2016, pp. 1–15.

[52] L. Long, W. Wang, J. Wen, M. Zhang, and Q. Lin, "Few-shot learning by exploiting object relation," in *Proc. Int. Conf. Learn. Represent. (ICLR)*, 2019.

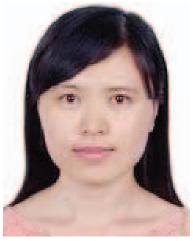
[53] N. Mishra, M. Rohaninejad, X. Chen, and P. Abbeel, "A simple neural attentive meta-learner," in *Proc. Int. Conf. Learn. Represent. (ICLR)*, 2018, pp. 1–20.

[54] V. G. Satorras and J. B. Estrach, "Few-shot learning with graph neural networks," in *Proc. Int. Conf. Learn. Represent. (ICLR)*, 2018.

[55] C. Simon, P. Koniusz, and M. Harandi, "Projective subspace networks for few-shot learning," in *Proc. Int. Conf. Learn. Represent. (ICLR)*, 2019.

Nan Lai received the B.E. degree in software engineering from the Huazhong University of Science and Technology, Wuhan, China, in 2015. She is currently pursuing the Ph.D. degree with the Institute of Computing Technology (ICT), Chinese Academy of Sciences (CAS), Beijing, China.

Her research mainly focuses on deep learning, few-shot learning, and meta-learning.



Meina Kan (Member, IEEE) received the B.S. degree in computer science from Shandong University, Jinan, China, in 2007, and the Ph.D. degree in computer vision from the Institute of Computing Technology (ICT), Chinese Academy of Sciences (CAS), Beijing, China, under the supervision of Prof. X. Chen and Prof. S. Shan, in 2013.

She is currently an Associate Professor with ICT, CAS. Her research mainly focuses on face recognition, transfer learning, multiview learning, and deep learning.

Dr. Kan has served as the Co-Chair for the ICPR18 Workshop on Deep Learning for Pattern Recognition and the ACCV14 Workshop on Human Identification for Surveillance.

Chunrui Han received the B.E. degree from the Ocean University of China (OUC), Qingdao, China, in 2016. She is currently pursuing the Ph.D. degree with the Institute of Computing Technology (ICT), Chinese Academy of Sciences (CAS), Beijing, China.

Her research interests include face recognition, few-shot learning, and meta-learning.

Xingguang Song, photograph and biography not available at the time of publication.

Shiguang Shan (Senior Member, IEEE) received the M.S. degree in computer science from the Harbin Institute of Technology, Harbin, China, in 1999, and the Ph.D. degree in computer science from the Institute of Computing Technology (ICT), Chinese Academy of Sciences (CAS), Beijing, China, in 2004.

After graduation, he joined ICT, CAS, where he became a Full Professor in 2010. He is currently the Deputy Director of the Key Laboratory of Intelligent Information Processing, CAS. He especially focuses on face recognition related research topics and machine learning with little or weakly supervised data. He is also personally interested in brain science, cognitive neuroscience, and their interdisciplinary research topics with AI. He has published more than 200 articles in refereed journals and proceedings in the areas of computer vision and pattern recognition. His research interests include computer vision, pattern recognition, and machine learning (deep learning).

Dr. Shan was a recipient of the China's State Natural Science Award in 2015 and the China's State S&T Progress Award in 2005. He has served as the Area Chair for many international conferences, including ICCV 2011, ICPR 2012/2014, ACCV 2012/2016/2018, FG 2013/2018, ICASSP 2014, and BTAS 2018. He is/was an Associate Editor of several international journals, including the IEEE TRANSACTIONS ON IMAGE PROCESSING, *Computer Vision and Image Understanding*, *Neurocomputing*, and *Pattern Recognition Letters*.