Cross-Domain Few-Shot Classification through Diversified Feature Transformation Layers

Li Yalan
Lanzhou University of Technology
Lanzhou, Gansu, China
e-mail: 1439127314@qq.com

Wu Jijie
Lanzhou University of Technology
Lanzhou, Gansu, China
e-mail: 1739055115@qq.com

2021 IEEE International Conference on Artificial Intelligence and Computer Applications (ICAICA) | 978-1-6654-1867-6/21/$31.00 ©2021 IEEE | DOI: 10.1109/ICAICA52286.2021.9498059

***Abstract*—The purpose of the few-shot classification is to classify new categories, and each category contains few labeled samples. The currently popular cross-domain few-shot classification uses a feature transformation layer to transform features to achieve the feature enhancement, so as to simulate various feature distributions in different domains during the training process. However, due to the large differences in the distribution of cross-domain features, a single feature transformation layer cannot perform multiple feature transformations. To obtain the change of the feature distribution in different domains, a diversified feature transformation is proposed based on the original feature transformation layer to solve the metric-based cross-domain few-shot classification problem Simulation results are obtained based on these five datasets commonly used in few-shot classification: mini-ImageNet, CUB, Cars, Places and Plantae. The simulation results show that the proposed diversified feature transformation layer can achieve good results in the metric-based model.**

Keywords—cross-domainf few-shot classification, feature enhancement, feature transformation layer

I. INTRODUCTION

The purpose of the few-shot [1] classification is to identify query samples in new categories, and each class contains few labeled support samples. Among the various latest methods to solve the few-shot classification problem, the metric-based meta-learning method [2] [3] has received a lot of attention due to its simplicity and effectiveness. The metric-based few-shot classification is based on the similarity between the query sample and the support sample for predictive classification. Among them, metric-based methods include the feature encoder (the feature extraction) and the metric function. Given a task that contains few labeled support samples and unlabeled query samples from a new class, the encoder first performs the feature extraction, and then inputs the support set and query set samples into the metric to clssify query samples. The metric-based few-shot classification can well classify new classes in the same field as the training phase. However, there exists a problem that the metric-based methods cannot classify samples from different fields well [4].

Many unsupervised adaptive technologies [5] [6] have been proposed to solve this problem. The unsupervised adaptation technology focuses on adapting the classifier of the same category from the source to the target domain. On the basis of the domain adaptation, a cross-domain knowledge transfer under relaxed constraints is proposed, so as to identify new classes at one time. The unsupervised method assumes that there are a large number of unlabeled images in the target domain during the training process, but in most cases this assumption cannot be realized. On the other hand, there have been many domain generalization methods [7] to learn classifiers, which can generalized well to multiple invisible domains without accessing these invisible data. However, the existing domain generalization methods are to only identify instances from the same category in the training process.

In order to improve the domain generalization ability of identifying new categories in the few-shot classification environment, a metric-based method is proposed in [8] to solve the domain drift the problem of few-shot classification. The authors use the feature transformation layer in the training process phase and apply the affine transformation to enhance image features, so as to simulate the distribution of features in different domains. In order to imitate the feature distribution in different domains, a meta-learning method is also proposed in [8] to learn the hyper-parameters of the feature transformation layer. However, due to the large differences in cross-domain features, a single feature transformation layer cannot perform the diversified feature transformation.

The singularity of the feature transformation layer based on a single feature transformation in different domains leads to a relatively single distribution of learned features. Based on this problem, we propose a diversified feature transformation layer.

Due to the significant differences in the features in different domains, the transformation features obtained by a feature transformation layer can more accurately represent the features of different domains more accurately when the feature transformation between the features of different domains in the training process is carried out. Hence, in the training process, we obtain multiple transformed features through multiple feature transformation layers in parallel, and average the multiple transformed features to obtain diversified transformation features.

Our contributions are divided into two aspects:

The multi-feature transformation layer is proposed to extract the features of various images from different fields. In addition, the proposed multiple feature transformation layer can be applied to various metric-based few-shot classification methods.

We have made improvements in three metric-based methods (RelationNet [3] MatchingNet [9] and Graph Neural Networks [2]). Simulation results show that our proposed metric-based diversified feature transformation layer can effectively improve the generalization ability of the model to the unseen domain.

II. Related work

A. Few-shot classification

Few-shot classification is the application of the meta-learning in the field of the supervised learning. Few-shot aims to learn to recognize novel categories, each with a limited number of labeled images. Few-shot has made a great progress in the field of the meta-learning. Currently, there are three popular meta-learning methods to solve few-shot classification problems, including the recursive framework [10], the optimization-based schemes [11] [12] and the metric-based classification methods [13] [4]. The recursive framework is proposed in [10], which process and encode the new categories of labeled images only once. The optimization-based schemes [11] [12] adjust the model by learning a few images in the meta-learning process. The metric-based classification methods [13] [4] classify the query samples by calculating the similarity between the query samples and a small number of labeled samples of the new class.

Among the above three types of methods, metric-based methods have attracted more attention because of their simplicity and effectiveness. The metric-based few-shot classification method mainly includes two steps. Step 1: the feature encoder performs encoding, and its main role is to extract features from the labeled and unlabeled images. Step 2: the similarity of the metric is measured, and its main role is to use the features output from the encoder as the input to obtain the category of unlabeled images. The more popular models include MatchingNet [23], ProtoNet [3] and GNN [2]. The MatchingNet [23] uses the cosine similarity and recursive networks together. The ProtoNet [3] the uses euclidean distance, the RelationNet [3] uses the CNN modules, and the GNN [2] uses the graph convolution module as a metric function. However, the features of images re-extracted from various tasks have relatively large distribution differences in various fields, which makes it difficult for these metrics to be extended to unseen fields. Recent studies have shown that the performance of the few-shot classification method decreases significantly in the case of domain shifts. Our work is to improve the generalization ability of the metric-based few-shot classification model in the unseen domain.

B. Domain adaptation

The purpose of the domain adaptation method [6] is to reduce the domain offset between the source domain and the target domain. Since the emergence of Domain Adversarial Neural Networks (DANN) [15], many frameworks apply the adversarial training to align the source and target [16] distributions or pixel level at the feature level[17] [18]. However, the goal of most domain frameworks is to apply the knowledge of the same domain learned from the source domain to the target domain, so the efficiency of classifying new classes in a few-shot environment is relatively low. But in [19] the problem of one-time domain transformation is solved. However, these domain-adaptive methods require access to unlabeled images in the target domain during the training process. It is difficult to collect images of rare categories at present, which makes the feasibility of this assumption relatively low.

C. Domain generalization

Different from the domain adaptation framework [2], the main role of the domain generalization method is to generalize a set of visible domains to invisible domains without visiting the unseen domains in the training process. Before the emergence of the meta-learning[20], several methods have been proposed to solve the problem of the domain generalization including the method that extracts invariant features from various seen domains[21], the method that improves the classifier by fusing classifiers learned from unseen domains[22], and the method that divides the classifier into domain-specific and domain-invariant components [23] [24]. On the other hand, the work is mainly to enhance the input data through the adversarial learning [25]. Many recent methods apply learning strategies to the generalization in the training process[26]. Different from these above methods, our work is mainly to obtain the conversion feature through the diversified feature conversion layer, and identify new categories in the invisible domain.

D. Learning-based data augmentation

The main purpose of the data augmentation method is to enhance the diversity of the data in the training process. The method of the data augmentation based on learning is different from the traditional manual method. At present, a variety of learning methods have been proposed to data augmentation [27] [28] [29]. For example, in the SmartAugmentation scheme [28], a network is trained to combine multiple images from the same category. The Bayesian DA [30] method to enhance the data according to the distribution learned from the training set. The core of the RenderGAN model is proposed in [29] which uses the generative confrontation model to continuously simulate and approximate the real image. In addition, the AutoAugment [27] algorithm enhances learning through the reinforcement learning. The two more popular framework methods [25] [26] are to model the differences in different fields through the adversarial learning to achieve the effect of the protective gear enhancement. Recently a feature transformation layer [14] is proposed in the training proposes, which enhances the features of the image through the affine transformation, and simulates the feature distribution in different domains. In order to capture the feature distribution in different domains, the author also uses the meta-learning to obtain the feature transformation.

E. Conditional normalization

The purpose of conditional normalization is to modulate the activation based on the radiation changes of learning based on the external data. Conditional normalization methods include Conditional Batch Normalization [31], adaptive normalization [32] and SPADE [33]. The above three methods are widely used the style transfor and the image synthesis task [34]. Besides the image stylization and generation, the conditional normalization is also used to align different data distributions for domain adaptation [35]. At present, the more popular TADAM method applies the conditional batch normalization to the metric-based model, which is used for the task of the few-shot classification. However, the main purpose of the TADAM method is to model the distribution of tasks in the same field, and it does not realize the modeling of the task distribution in cross-domain situations. Compared with the TADAM method, our method can model the distribution of pictures in different fields.

III. Diversified Feature Transformation Layer

A. Preliminaries

Few-shot classification problems are usually characterized by N_W way (the number of categories) and N_S shot (the number of labeled samples in each category). The metric-based algorithm usually includes a feature encoder E and a metric function M. During each iteration of the training process, the algorithm randomly samples N_W categories as a task T. We denote the set of input images as $X=\{x_1,x_2,\cdots,x_n\}$ and the corresponding class label as $X=\{x_1,x_2,\cdots,x_n\}$. Task T consists of the support set C $S=\{(X_S,Y_S)\}$ and the query set $Q=\{(X_q,Y_q)\}$. N_S and N_q samples are randomly selected for each of the N_W categories to form the support set S and the query set Q respectively. Firstly, the feature encoder E performs the feature extraction on the images of the support set and the images of the query set, and then the features $E(x^q)$ from the query sample and the features $E(X_S)$ from the labeled support sample are sent to the metric function to predict the query image.The above process can be expressed as:

$$\hat{Y}=M(Y_S,E(X_S),E(X_q)) \tag{1}$$

The training goal is to use the classification loss of the images in the query set to optimize the framework. The loss function is as follows:

$$L=L_{cls}(Y_q,\hat{Y}_q) \tag{2}$$

There are big differences between various metric-based algorithms. Typical metric-based methods are MatchingNet, RelationNet and GNN. Among them, the MatchingNet framework uses the long-short-term memory (LSTM)cto measure similarity. The RelationNet model uses a convolutional neural network for similarity measurement, while GNN uses a graph convolutional network to measure the similarity. In this work, we want to solve the problem of singularization of the feature transformation layer in the domain generalization. We represent a domain as a set of few-shot classification problems, and we can represent it as $\mathrm{T}=\{\mathrm{T}_1,\mathrm{T}_2,\cdots,\mathrm{T}_n\}$. In this task, our goal is that the metric-based few-shot classification model learned in the seen domain can be well generalized to the unseen domain T^{unseen}.

B. Diversified feature conversion layer

In this work, our focus is to improve the generalization ability of the few-shot model based on the metric in the invisible domain. Due to the large differences in the features extracted in the tasks of the seen and unseen domains, the metric may be excessively suitable for the seen domain and cannot be extended to the unseen domain. Figure 1 shows the network structure of our proposed diversified feature transformation layer.

In the feature transformation layer, we gather the changes of the feature together, and activate the extended feature encoder with intermediate features with radial changes. That is to say, the feature encoder integrated with the diversified feature feature transformation layer can generate more kinds of feature distributions, thereby further improving the generalization ability of the metric. In the experiment, we insert a diversified feature transformation layer after each batch normalization. The hyper-parameters $\theta^n_{\gamma^n}\in\Re^{C\times1\times1}$ and $\theta^n_{\beta^n}\in\Re^{C\times1\times1}$ in the diversified feature transformation layer represent the standard deviation of the

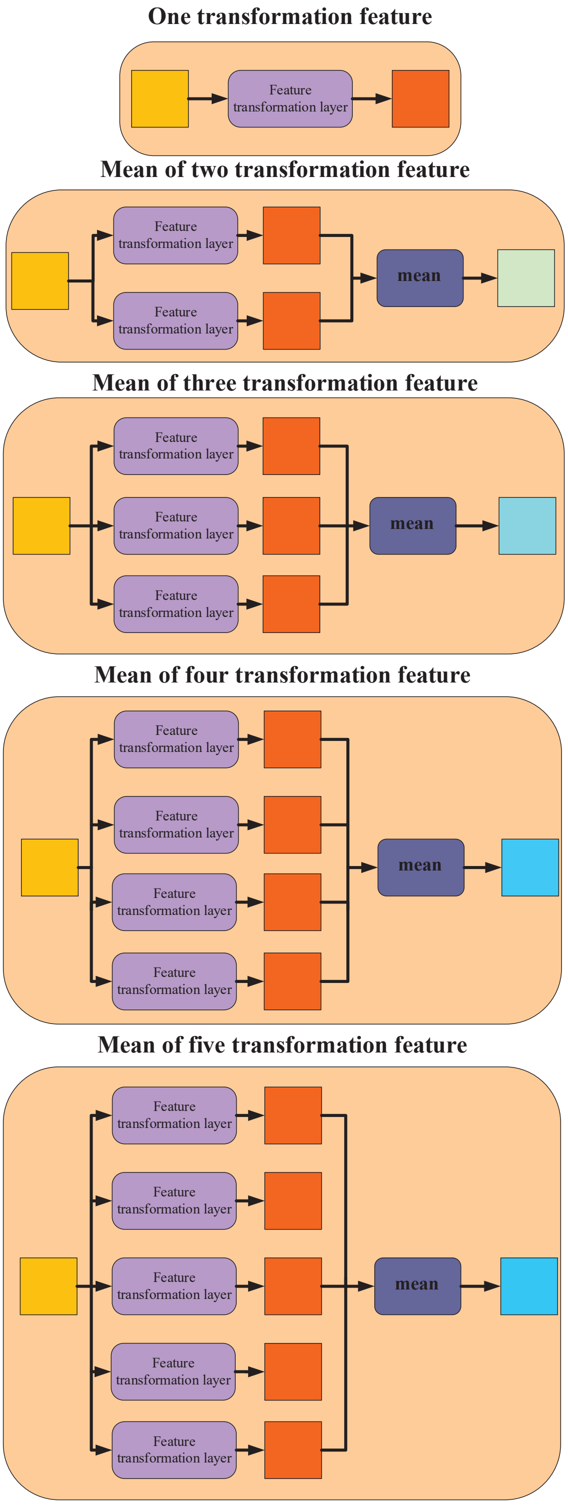

Figure 1: Network structure of 1 to 5 feature transformation layers.

Gaussian distribution used for the affine transformation parameter sampling, where n is the parallel n-th feature transformation layer. When the activation feature dimension of the middle image z in the given feature encoder is

$C \times H \times W$, we obtain the weight γ^n and the bias β^n from the Gaussian distribution, which are:

$$\gamma^n \sim \mathrm{N}(1, softplus(\theta^n_{\gamma^n})) \quad \beta^n \sim \mathrm{N}(0, softplus(\theta^n_{\beta^n})) \tag{3}$$

then the modulated activation of each feature transformation layer is calculated as:

$$\hat{z}^n_{c,h,w} = \gamma^n_c \times z_{c,h,w} + \beta^n_c \tag{4}$$

Finally, the modulated activation calculation is:

$$\hat{z}_{c,h,w} = \hat{z}^1_{c,h,w} + \hat{z}^1_{c,h,w} + \cdots + \hat{z}^n_{c,h,w} \tag{5}$$

where $\hat{z}^n_{c,h,w} \in \hat{Z}^n$ and $z_{c,h,w} \in Z$.

In this experiment, we have a detailed description of the algorithm structure. In each iteration of the training process, we sample a pseudo-seen domain T^{PS} and a pseudo-unseen domain T^{PU} from a set of seen domains $\{T^{seen}_1,\ T^{seen}_2, \cdots, T^{seen}_N\}$. Given a metric-based model, including the feature encoder $E_{\theta^t_e}$ and the metric function $M_{\theta^t_m}$. We first integrate the layer with hyper-parameter $\theta^t_f = \{\theta^{1t}{}_\gamma \theta^{2t}{}_\gamma, \cdots, \theta^{nt}{}_\gamma; \theta^{1t}{}_\beta, \theta^{2t}{}_\beta, \cdots, \theta^{nt}{}_\beta\}$ into the feature encoder, which is expressed as $E_{\theta^T_E, \theta^T_F}$. Then the parameters $(\theta^{t+1}_e, \theta^{t+1}_m)$ in the model is updated based on the loss function (2), which are:

$$\begin{aligned}(\theta^{t+1}_e, \theta^{t+1}_m) &= (\theta^t_e, \theta^t_m) - \\ &\alpha \nabla \theta^t_e, \theta^t_m L_{cls}(Y^{ps}_q, M_{\theta^T_M}(Y^{ps}_s, E_{\theta^t_e, \theta^t_f}(X^{ps}_s), E_{\theta^t_e, \theta^t_f}(X^{ps}_q))),\end{aligned} \tag{6}$$

where α is the learning rate. We delete the diversified feature transformation layer from the model, and then calculate the classification loss of the updated model on the pseudo-unseen domain $T^{\mathrm{pu}} = \{(X^{pu}_s, Y^{pu}_s), (X^{pu}_q, Y^{pu}_q)\} \in T^{pu}$. The classification loss is obtained as:

$$L^{\mathrm{pu}} = L_{cls}(Y^{pu}_q, M_{\theta^{t+1}_m}(Y^{pu}_s, E_{\theta^{t+1}_e}(X^{pu}_s), E_{\theta^{t+1}_e}(X^{pu}_q))) \tag{7}$$

Finally, since the loss L^{pu} reflects the effectiveness of the diversified feature transformation layer, we use the following methods to optimize the hyper-parameter θ_f is optimized as:

$$\theta^{t+1}_f = \theta^t_f - \alpha \nabla \theta^t_f L^{pu} \tag{8}$$

Note that our proposed diversified feature transformation layer and metric-based model are jointly optimized during the training process.

IV. Experimental result

We use three metric-based methods to verify the effectiveness of our proposed diversified feature transformation layer. We train a metric-based few-shot classification model on the mini-ImageNet [36] domain, and test the trained model on four different domains. The four different domains are CUB [37] , Cars [38] , Places [39] and Plantae [40]. In addition, in order to verify the generalization performance of the proposed diversified feature transformation layer, we selected one of the CUB, Cars, Places and Plantae domains as the unseen domain, mini-ImageNet and other domains as the seen domain for training. Then the selected unseen domain is used to test.

Before the few-shot classification training, we first use the standard cross-entropy loss function of 64 training categories in the mini-ImageNet dataset to pre-train the feature encoder E. In the pre-training and later training stages, we use the public model provided by [4] to train the MatchingNet and RelationNet models.

A. 5way5shot for multiple domains

Table I shows the experimental results of 5way5shot. In the experiment, we use the leave-one-out method to select use one of the four datasets of CUB, Cars, Places and Plantae as the unseen domain, the other datasets and the mini-ImageNet dataset as the seen domain. Among them, two feature transformation layers (ours), three feature transformation layers (ours), four feature transformation layers (ours) and five feature transformation layers (ours) are the proposed diversified feature transformation layers. We average the transformation features obtained by multiple feature transformation layers to obtain diversified transformation features. The proposed diversified feature transformation layer is implemented on three metric-based (MatchingNe, RelationNet, GNN) frameworks. During the experiment, we increase the number of feature transformation layers up to 5 to get the experimental results. It can be seen from Table I that the proposed diversified feature transformation layer improves the classification accuracy to a certain extent. The experimental results show that our proposed feature transformation layer can better capture the changes in feature distribution between different domains, and can improve the generalization ability of feature transformation. The proposed diversified feature transformation layer is optimized through learning, and the feature transformation layer can more extensively learn the feature distribution in the unseen domain.

B. 5way1shot for multiple domains

In order to further prove the effect of the proposed diversified feature transformation layer, we also use the leave-one-out method to select each data set as the unseen domain, and the other data sets as the seen domain. Set the parameter to 5way1shot for the experimentation. The results are shown in Table II. As shown in Table II, the proposed diversified feature transformation layer also achieves good results on 5 way 1 shot.

C. Single domain

For the fairness of the experiment, we also use the mini-ImageNet dataset as an unseen domain to train the model on 5way5shot, and use the dataset CUB, Cars, Places and Plantae as the unseen domain to test the experimental results as shown in Table III. The results show that our proposed diversified feature transformation layer can also improve the classification progress when the mini-ImageNet dataset is invisible. This shows that the proposed diversified feature transformation layer can better capture the changes in feature distribution between different domains. The diversified feature transformation layer can better learn the features of the unseen domain.

TABLE I THE EXPERIMENTAL RESULTS OF 5WAY5SHOT USING THE LEAVE-ONE-OUT METHOD TO SET THE INVISIBLE DOMAIN AND TRAIN THE MODEL.

Unseen domain	Feature transformation layer	MatchingNet	RelationNet	GNN
cars	One feature transformation layers	35.42 ± 0.52	41.07 ± 0.53	47.99 ± 0.66
	Two feature transformation layers(ours)	**40.34 ± 0.55**	**42.35 ± 0.56**	**48.57 ± 0.69**
	Three feature transformation layers(ours)	31.03 ± 0.51	41.01 ± 0.53	**50.32 ± 0.69**
	Four feature transformation layers(ours)	**36.66 ± 0.54**	**43.41 ± 0.57**	**48.91 ± 0.68**
	Five feature transformation layers(ours)	**35.58 ± 0.52**	**42.14 ± 0.55**	**49.55 ± 0.68**
places	One feature transformation layers	54.19 ± 0.59	64.29 ± 0.58	69.23 ± 0.67
	Two feature transformation layers(ours)	**54.35 ± 0.61**	62.07 ± 0.58	68.30 ± 0.65
	Three feature transformation layers(ours)	53.30 ± 0.57	62.70 ± 0.55	**72.74 ± 0.65**
	Four feature transformation layers(ours)	53.04 ± 0.67	62.16 ± 0.57	67.31 ± 0.67
	Five feature transformation layers(ours)	53.80 ± 0.59	61.79 ± 0.57	**69.38 ± 0.66**
cub	One feature transformation layers	48.76 ± 0.56	64.15 ± 0.55	71.12 ± 0.68
	Two feature transformation layers(ours)	45.21 ± 0.55	62.95 ± 0.54	**71.33 ± 0.65**
	Three feature transformation layers(ours)	**48.94 ± 0.56**	**64.20 ± 0.55**	**71.52 ± 0.68**
	Four feature transformation layers(ours)	45.03 ± 0.52	**64.47 ± 0.54**	**72.18 ± 0.68**
	Five feature transformation layers(ours)	44.73 ± 0.59	64.03 ± 0.52	69.75 ± 0.66
plantae	One feature transformation layers	46.27 ± 0.52	51.20 ± 0.55	60.68 ± 0.67
	Two feature transformation layers(ours)	**50.21 ± 0.57**	51.33 ± 0.55	59.17 ± 0.70
	Three feature transformation layers(ours)	45.66 ± 0.50	50.23 ± 0.54	**60.86 ± 0.66**
	Four feature transformation layers(ours)	44.19 ± 0.52	50.64 ± 0.56	59.17 ± 0.64
	Five feature transformation layers(ours)	48.23 ± 0.52	**51.73 ± 0.55**	58.76 ± 0.64

TABLE II THE EXPERIMENTAL RESULTS OF 5WAY1SHOT USING THE LEAVE-ONE-OUT METHOD TO SET THE UNSEEN DOMAIN AND TRAIN THE MODEL.

Unseen domain	Feature transformation layer	MatchingNet	RelationNet	GNN
cars	One feature transformation layers	29.40 ± 0.47	31.75 ± 0.51	33.51 ± 0.58
	Two feature transformation layers(ours)	**30.53 ± 0.52**	**31.85 ± 0.49**	32.63 ± 0.56
	Three feature transformation layers(ours)	25.56 ± 0.38	31.12 ± 0.48	32.34 ± 0.54
	Four feature transformation layers(ours)	**30.92 ± 0.51**	**33.00 ± 0.53**	**33.54 ± 0.60**
	Five feature transformation layers(ours)	29.02 ± 0.49	30.92 ± 0.47	**33.73 ± 0.59**
places	One feature transformation layers	42.03 ± 0.60	52.46 ± 0.66	52.96 ± 0.79
	Two feature transformation layers(ours)	41.22 ± 0.59	48.64 ± 0.64	49.38 ± 0.74
	Three feature transformation layers(ours)	40.53 ± 0.58	49.43 ± 0.66	48.82 ± 0.72
	Four feature transformation layers(ours)	40.17 ± 0.51	43.20 ± 0.59	51.03 ± 0.74
	Five feature transformation layers(ours)	41.67 ± 0.62	44.89 ± 0.58	51.81 ± 0.73
cub	One feature transformation layers	37.61 ± 0.55	46.26 ± 0.59	49.69 ± 0.73
	Two feature transformation layers(ours)	35.27 ± 0.53	45.49 ± 0.57	**50.22 ± 0.70**
	Three feature transformation layers(ours)	**37.81 ± 0.55**	44.49 ± 0.59	47.84 ± 0.69
	Four feature transformation layers(ours)	35.87 ± 0.53	**46.87 ± 0.62**	48.07 ± 0.73
	Five feature transformation layers(ours)	36.69 ± 0.42	**46.44 ± 0.61**	46.15 ± 0.68
plantae	One feature transformation layers	59.39 ± 0.50	36.16 ± 0.54	39.65 ± 0.63
	Two feature transformation layers(ours)	**61.58 ± 0.51**	35.48 ± 0.56	**39.76 ± 0.62**
	Three feature transformation layers(ours)	58.79 ± 0.48	**37.63 ± 0.56**	**41.66 ± 0.66**
	Four feature transformation layers(ours)	55.48 ± 0.59	35.62 ± 0.54	38.94 ± 0.62
	Five feature transformation layers(ours)	57.80 ± 0.52	35.57 ± 0.54	**39.97 ± 0.60**

TABLE III THE 5 WAY 5 SHOT EXPERIMENT RESULTS OF SETTING THE MINI-IMAGENET IN THE UNSEEN DOMAIN AND TRAINING THE MODEL.

Unseen domain	Feature transformation layer	MatchingNet	RelationNet	GNN
cars	One feature transformation layers	39.11 ± 0.52	43.86 ± 0.56	46.51 ± 0.64
	Two feature transformation layers(ours)	37.47 ± 0.57	**43.98 ± 0.58**	**48.85 ± 0.67**
	Three feature transformation layers(ours)	36.14 ± 0.55	43.24 ± 0.56	**48.46 ± 0.72**
	Four feature transformation layers(ours)	37.22 ± 0.55	41.93 ± 0.57	**48.04 ± 0.70**
	Five feature transformation layers(ours)	35.37 ± 0.54	41.97 ± 0.54	**47.91 ± 0.64**
places	One feature transformation layers	53.55 ± 0.57	59.56 ± 0.56	69.64 ± 0.69
	Two feature transformation layers(ours)	**53.76 ± 0.55**	**64.74 ± 0.57**	67.83 ± 0.66
	Three feature transformation layers(ours)	52.87 ± 0.58	**63.01 ± 0.56**	67.35 ± 0.69
	Four feature transformation layers(ours)	52.02 ± 0.60	**64.77 ± 0.55**	**74.09 ± 0.66**
	Five feature transformation layers(ours)	51.72 ± 0.56	**63.49 ± 0.55**	69.12 ± 0.67
cub	One feature transformation layers	51.34 ± 0.54	63.92 ± 0.57	71.72 ± 0.66
	Two feature transformation layers(ours)	47.47 ± 0.54	61.30 ± 0.56	70.04 ± 0.67
	Three feature transformation layers(ours)	**52.81 ± 0.56**	**64.61 ± 0.58**	68.95 ± 0.66
	Four feature transformation layers(ours)	**53.01 ± 0.54**	**64.33 ± 0.58**	**71.67 ± 0.67**
	Five feature transformation layers(ours)	45.50 ± 0.57	62.83 ± 0.56	69.35 ± 0.66
plantae	One feature transformation layers	40.31 ± 0.50	52.49 ± 0.56	58.33 ± 0.64
	Two feature transformation layers(ours)	**48.19 ± 0.54**	49.71 ± 0.54	55.65 ± 0.67
	Three feature transformation layers(ours)	**43.20 ± 0.54**	50.54 ± 0.54	**59.54 ± 0.63**
	Four feature transformation layers(ours)	**43.57 ± 0.53**	49.37 ± 0.53	57.43 ± 0.66
	Five feature transformation layers(ours)	**46.02 ± 0.57**	51.97 ± 0.53	58.12 ± 0.65

V. CONCLUSION

We propose a few-shot classification framework based on the metric-based diversified feature transformation. Our core idea is to use the diversified feature transformation layer to simulate the distribution of feature changes in different domains. A large number of experiments show that our method is more accurate than a single feature transformation layer to learn the distribution of feature changes in different domains. Our algorithm is a good framework for the few-shot classification to solve the feature transformation in different domains.

REFERENCES

[1] Brenden M Lake, Ruslan Salakhutdinov, and Joshua B Tenenbaum. Human-level concept learning through probabilistic program induction. Science, 350(6266):1332–1338, 2015.

[2] Victor Garcia and Joan Bruna. Few-shot learning with graph neural networks. In ICLR, 2018.

[3] Flood Sung, Yongxin Yang, Li Zhang, Tao Xiang, Philip HS Torr, and Timothy M Hospedales.Learning to compare: Relation network for few-shot learning. In CVPR, 2018.

[4] Wei-Yu Chen, Yen-Cheng Liu, Zsolt Kira, Yu-Chiang Wang, and Jia-Bin Huang. A closer look at few-shot classification. In ICLR, 2019a.

[5] Sinno Jialin Pan and Qiang Yang. A survey on transfer learning. IEEE Transactions on Knowledge and Data Engineering, 22(10):1345–1359, 2010.

[6] Yuhua Chen, Wen Li, Christos Sakaridis, Dengxin Dai, and Luc Van Gool. Domain adaptive faster r-cnn for object detection in the wild. In CVPR, 2018.

[7] Gilles Blanchard, Gyemin Lee, and Clayton Scott. Generalizing from several related classifiification tasks to a new unlabeled sample. In NIPS, 2011.

[8] Cross-Domain Few-Shot Classification VIA Learned Feature-Wise Transformation.in ICLR, 2020.

[9] Oriol Vinyals, Charles Blundell, Tim Lillicrap, Daan Wierstra, et al. Matching networks for one shot learning. In NIPS, 2016.

[10] Danilo Jimenez Rezende, Shakir Mohamed, Ivo Danihelka, Karol Gregor, and Daan Wierstra. One-shot generalization in deep generative models. JMLR, 48, 2016.

[11] Chelsea Finn, Pieter Abbeel, and Sergey Levine. Model-agnostic meta-learning for fast adaptation of deep networks. In ICML, 2017.

[12] Andrei A Rusu, Dushyant Rao, Jakub Sygnowski, Oriol Vinyals, Razvan Pascanu, Simon Osindero, and Raia Hadsell. Meta-learning with latent embedding optimization. In ICLR, 2019.

[13] Gregory Koch, Richard Zemel, and Ruslan Salakhutdinov. Siamese neural networks for one-shot image recognition. In ICML Workshop, 2015.

[14] Jake Snell, Kevin Swersky, and Richard Zemel. Prototypical networks for few-shot learning. In NIPS, 2017.

[15] Yaroslav Ganin, Evgeniya Ustinova, Hana Ajakan, Pascal Germain, Hugo Larochelle, Franc¸ois Laviolette, Mario Marchand, and Victor Lempitsky. Domain-adversarial training of neural networks. JMLR, 17(1):2096–2030, 2016.

[16] Han-Kai Hsu, Chun-Han Yao, Yi-Hsuan Tsai, Wei-Chih Hung, Hung-Yu Tseng, Maneesh Singh, and Ming-Hsuan Yang. Progressive domain adaptation for object detection. In WACV, 2020.

[17] Yi-Hsuan Tsai, Wei-Chih Hung, Samuel Schulter, Kihyuk Sohn, Ming-Hsuan Yang, and Manmohan Chandraker. Learning to adapt structured output space for semantic segmentation. In CVPR, 2018.

[18] Judy Hoffman, Eric Tzeng, Taesung Park, Jun-Yan Zhu, Phillip Isola, Kate Saenko, Alexei A. Efros, and Trevor Darrell. Cycada: Cycle consistent adversarial domain adaptation. In ICML, 2018.

[19] Nanqing Dong and Eric P Xing. Domain adaption in one-shot learning. In Joint European Conference on Machine Learning and Knowledge Discovery in Databases, 2018.

[20] Sachin Ravi and Hugo Larochelle. Optimization as a model for few-shot learning. In ICLR, 2017.

[21] Haoliang Li, Sinno Jialin Pan, Shiqi Wang, and Alex C Kot. Domain generalization with adversarial feature learning. In CVPR, 2018b.

[22] Li Niu, Wen Li, and Dong Xu. Multi-view domain generalization for visual recognition. In ICCV, 2015a.

[23] Aditya Khosla, Tinghui Zhou, Tomasz Malisiewicz, Alexei A Efros, and Antonio Torralba. Undoing the damage of dataset bias. In ECCV, 2012.

[24] Da Li, Yongxin Yang, Yi-Zhe Song, and Timothy M Hospedales. Deeper, broader and artier domain generalization. In ICCV, 2017a.

[25] Shiv Shankar, Vihari Piratla, Soumen Chakrabarti, Siddhartha Chaudhuri, Preethi Jyothi, and Sunita Sarawagi. Generalizing across domains via cross-gradient training. In ICLR, 2018.

[26] Yogesh Balaji, Swami Sankaranarayanan, and Rama Chellappa. Metareg: Towards domain generalization using meta-regularization. In NeurIPS, 2018.

[27] Ekin D Cubuk, Barret Zoph, Dandelion Mane, Vijay Vasudevan, and Quoc V Le. Autoaugment: Learning augmentation policies from data. In CVPR, 2019.

[28] Leon Sixt, Benjamin Wild, and Tim Landgraf. Rendergan: Generating realistic labeled data. Frontiers in Robotics and AI, 5:66, 2018.

[29] Toan Tran, Trung Pham, Gustavo Carneiro, Lyle Palmer, and Ian Reid. A bayesian data augmentation approach for learning deep models. In NIPS, 2017.

[30] Vincent Dumoulin, Jonathon Shlens, and Manjunath Kudlur. A learned representation for artistic style. In ICLR, 2017.

[31] Vincent Dumoulin, Jonathon Shlens, and Manjunath Kudlur. A learned representation for artistic style. In ICLR, 2017.

[32] Xun Huang and Serge Belongie. Arbitrary style transfer in real-time with adaptive instance normalization. In ICCV, 2017.

[33] Taesung Park, Ming-Yu Liu, Ting-Chun Wang, and Jun-Yan Zhu. Semantic image synthesis with spatially-adaptive normalization. In CVPR, 2019.

[34] Badour AlBahar and Jia-Bin Huang. Guided image-to-image translation with bi-directional feature transformation. In ICCV, 2019.

[35] Fabio Maria Cariucci, Lorenzo Porzi, Barbara Caputo, Elisa Ricci, and Samuel Rota Bulo. Autodial: Automatic domain alignment layers. In ICCV, 2017.

[36] Konstantinos Bousmalis, Nathan Silberman, David Dohan, Dumitru Erhan, and Dilip Krishnan. Unsupervised pixel-level domain adaptation with generative adversarial networks. In CVPR, 2017

[37] P. Welinder, S. Branson, T. Mita, C. Wah, F. Schroff, S. Belongie, and P. Perona. Caltech-UCSD Birds 200. Technical Report CNS-TR-2010-001, California Institute of Technology, 2010.

[38] Jonathan Krause, Michael Stark, Jia Deng, and Li Fei-Fei. 3d object representations for fifine-grained categorization. In 4th International IEEE Workshop on 3D Representation and Recognition, 2013.

[39] Bolei Zhou, Agata Lapedriza, Aditya Khosla, Aude Oliva, and Antonio Torralba. Places: A 10 million image database for scene recognition. TPAMI, 2017.

[40] Grant Van Horn, Oisin Mac Aodha, Yang Song, Yin Cui, Chen Sun, Alex Shepard, Hartwig Adam, Pietro Perona, and Serge Belongie. The inaturalist species classifification and detection dataset. In CVPR, 2018