IEEE Access
Multidisciplinary ⋮ Rapid Review ⋮ Open Access Journal

Received December 1, 2021, accepted February 8, 2022, date of publication February 11, 2022, date of current version February 17, 2022.

Digital Object Identifier 10.1109/ACCESS.2022.3151057

Improving Augmentation Efficiency for Few-Shot Learning

WONHEE CHO AND EUNWOO KIM, (Member, IEEE)
School of Computer Science and Engineering, Chung-Ang University, Seoul 06974, South Korea

Corresponding author: Eunwoo Kim (eunwoo@cau.ac.kr)

This work was supported in part by the Chung-Ang University Research Scholarship Grants in 2020; and in part by the National Research Foundation of Korea (NRF) grant funded by the Korea Government Ministry of Science and ICT (MSIP) [No. NRF-2020R1A4A4078916].

ABSTRACT While human intelligence can easily recognize some characteristics of classes with one or few examples, learning from few examples is a challenging task in machine learning. Recently emerging deep learning generally requires hundreds of thousands of samples to achieve generalization ability. Despite recent advances in deep learning, it is not easy to generalize new classes with little supervision. Few-shot learning (FSL) aims to learn how to recognize new classes with few examples per class. However, learning with few examples makes the model difficult to generalize and is susceptible to overfitting. To overcome the difficulty, data augmentation techniques have been applied to FSL. It is well-known that existing data augmentation approaches rely heavily on human experts with prior knowledge to find effective augmentation strategies manually. In this work, we propose an efficient data augmentation network, called EDANet, to automatically select the most effective augmentation approaches to achieve optimal performance of FSL without human intervention. Our method overcomes the disadvantages of relying on domain knowledge and requiring expensive labor to design data augmentation rules manually. We demonstrate the proposed approach on widely used FSL benchmarks (Omniglot and mini-ImageNet). The experimental results using three popular FSL networks indicate that the proposed approach improves performance over existing baselines through an optimal combination of candidate augmentation strategies.

INDEX TERMS Few-shot learning, automatic search, efficient augmentation.

I. INTRODUCTION

Over the past decade, we have witnessed remarkable performance gain with deep learning in many tasks, such as classification [2], [3], detection [4], [5], and segmentation [6], [7]. The extensive calculation of deep learning through artificial neural networks, advances in computing power, and numerous labeled examples have allowed deep learning models to achieve performance improvement. However, many real-world scenarios do not allow us to access sufficient labeled data due to some reasons, including privacy, security, high labeling costs, and difficulty managing data. Therefore, many researchers have attempted to learn deep learning algorithms with few examples, and the field of few-shot learning (FSL) [17], [18] has been recently emerged.

Few-shot learning recognizes patterns in data with few examples. In general, FSL approaches can be divided into two major streams; 1) how to compensate for insufficient data by adding supporting data [22], [23] and 2) how to represent a large parameter space covered by few training samples [25], [26]. The FSL approaches can be further categorized into four families: metric-based [19]–[21], data augmentation-based [22]–[24] optimization-based [25], [26], and semantic-based approaches [27], [28].

In this work, we address both metric-based and data augmentation-based FSL approaches. Among them, metric-based FSL has been actively studied, and the matching networks (MatchingNet) [19], prototypical networks (PrototypicalNet) [20], and relation networks (RelationNet) [21] have been popularly used. Note that it is important to learn an appropriate metric space for the approaches. One can learn the similarity between two samples, extract features from the samples, and calculate the distance between the features.

MatchingNet compares the cosine distance between samples mapped in embedding space, and PrototypicalNet computes the Euclidean distances between prototype samples

The associate editor coordinating the review of this manuscript and approving it for publication was Jinjia Zhou.

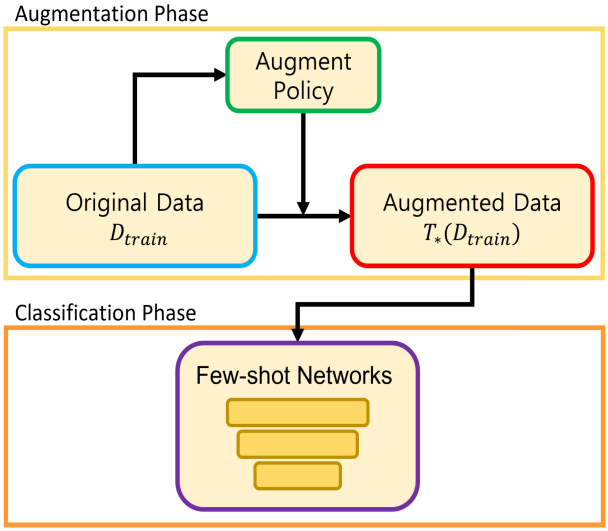

FIGURE 1. Overview of the proposed EDANet. In the augmentation phase, we automatically find the most efficient augmentation policy from the original data $\mathcal{D}_{train}$ to produce the augmented data $\mathcal{T}_*(\mathcal{D}_{train})$. In the classification phase, the FSL network is trained with the augmented training data, and then the performance is evaluated using the original test dataset.

representing classes. In addition, RelationNet introduces a relation metric to compare relationships between samples or between classes. Even if the metric-based FSL algorithms made an important contribution to early FSL, they are still unsatisfying compared to traditional many-shot learning approaches in terms of task performance. Commonly, data augmentation is a de facto practical technique for improving task performance. Therefore, this study aims to find an optimal data augmentation strategy from a pool of candidate augmentation techniques for metric-based FSL.

In the data augmentation-based approach, a few available samples are augmented to generate diverse samples to enrich the training experience. Data augmentation methods provide additional training data by transforming existing data samples into supplementary ones. The data augmentation-based FSL approaches [22]–[24] augment data by hallucinating the feature vectors from the training dataset. IDeMeNet [24] generates synthesized examples from the data augmentation method. However, depending on the domain, data augmentation methods have different strategies to improve task performance. It is important to note that such augmentation-based algorithms generate training samples via hand-crafted data augmentation rules, which play an important role in improving performance. However, those approaches rely on expert domain knowledge and require high-cost labor. Moreover, optimal augmentation rules can be problem-specific, making them difficult to apply to other tasks. Therefore, manually designed data augmentation strategies may produce sub-optimal solutions.

We overcome the difficulty of manual design choice in augmentation-based FSL. Focusing on this point, we first apply various augmentation methods to the metric-based FSL algorithms. Since most of the existing FSL algorithms have manually applied the augmentation method, we propose an Efficient Data Augmentation method, termed EDANet, that automatically searches for an optimal augmentation rule. It can provide an optimal combination of candidate augmentation methods that help enrich the network knowledge. Figure 1 provides an overview of EDANet. It consists of two phases: the augmentation and classification phases. In the augmentation phase, we explore data augmentation policies that improve performance. We use a density matching algorithm [30] to find the best combination of data augmentation strategies automatically. Afterward, we evaluate the performance of metric-based FSL networks with the augmented dataset in the classification phase.

The proposed approach reduces the manual design efforts, improves the quality of augmented data, and enhances the generalization ability of the FSL models. It also automatically searches for augmentation policies such as augmentation type and magnitude. To our knowledge, this is the first work to automate the search mechanism for the augmentation-based FSL.

We apply the proposed method to a range of metric-based FSL methods and evaluate them using image classification benchmark datasets. We use the Omniglot [18] and mini-ImageNet [19] datasets as benchmarks and analyze the performance of the approaches in terms of different distance metrics. Experimental results show that EDANet improves the performance by providing an optimal combination of candidate data augmentation techniques. Specifically, EDANet significantly improves the classification accuracy over the baseline methods (i.e. MatchingNet [19], PrototypicalNet [20], and RelationNet [21]) on mini-ImageNet, achieving 63.51% one-shot accuracy and 79.74% five-shot accuracy. Our approach achieves 98.61% one-shot accuracy and 99.13% five-shot accuracy on Omniglot, outperforming existing augmentation-based FSL baselines. By modifying the backbone architecture to ResNet-18 and ResNet-50, EDANet further improves the classification task by 5.67% on the Omniglot dataset and 1.97% on mini-ImageNet for one-shot accuracy, respectively.

II. RELATED WORK

A. FEW-SHOT LEARNING

Few-shot learning (FSL) aims to learn a model with a limited amount of labeled examples for predicting novel classes. FSL is also known as n-way k-shot learning, where n and k denote the number of classes and the number of data points per class, respectively. Each sub-task consisting of n-way k-shot is called an episode (mini-batch), which is why FSL is called episodic learning. A query and a support set are used for episodic training, where the support set is used to learn to solve a classification task, and the query set is used to evaluate the performance of the task. The major challenge of FSL is the insufficient supply of training examples, assuming that massive amounts of datasets are expensive to label correctly. Thus, FSL tries to improve the prediction capability and generalization performance of a

model with limited datasets. The FSL approaches can be classified into four categories: metric-based [19]–[21], data augmentation-based [22]–[24], optimizer-based [25], [26], and semantic-based approaches [27], [28]. Among them, we address metric-based and data augmentation-based approaches that are closely related to this work.

Metric-based approaches [19]–[21] learn a representation of the data with a metric in the feature space. These approaches transform data into a low-dimensional subspace, cluster the transformed samples, and compare the clusters using a metric function. MatchingNet [19] is one of the popular metric-based methods that compute the cosine distance to classify query samples. This method proposes a full context encoding module, which conditions the weights on the whole support set across classes. PrototypicalNet [20] computes the average features, called prototypes, for each class in the support set. Then, it classifies query samples by calculating the Euclidean distance between each prototype and the samples in the query set. RelationNet [21] classifies samples of novel classes by calculating the relation score between the query and sample for each novel class.

B. DATA AUGMENTATION

Data augmentation is a practical strategy used for a variety of learning-based tasks. Random cropping, rotating, clipping, flipping, scaling, and color transform are used as baseline augmentation methods [1], [2], [16] when applying them to image datasets, such as CIFAR [10] and ImageNet [9]. Mixup [12] generates an augmented image by mixing two images using linear interpolation. Cutout [13] is a regional dropout strategy in which random patches are replaced with zeros (black pixels). Recently, CutMix [14] was proposed to cut and paste part of a randomly selected image into another image. Such augmentation techniques have been used in supervised learning.

Note that the chronic problem of FSL is that there are not enough training samples. Since FSL is a data-hungry method, it also deploys data augmentation approaches [22]–[24]. SGM and PMN [22], [23] generate additional samples from trained hallucinators. Based on this, they propose strategies to improve the performance of FSL by using the generated samples. In addition, IDeMeNet [24] has adaptively fused samples from the support set to generate synthesized samples. This method trains an embedding submodule, which maps samples to feature representations and performs FSL.

However, the aforementioned approaches rely on hand-crafted rules and require task-specific domain knowledge and cumbersome exploration to find optimal augmentation strategies. We break away from the augmentation techniques that have been routinely applied in the existing metric-based FSL. In our work, we do not generate synthetic samples but explore promising augmentation techniques to find an optimal augmentation strategy. As such, we do not rely on hand-crafted rules and require task-specific knowledge to find the optimal augmentation strategy. Finally, we propose an efficient automatic augmentation method for FSL.

C. AUTOMATED LEARNING

One promising way is to find data augmentation methods automatically [29]–[32]. Recently, automating the design process of a neural network, called neural architecture search (NAS) [8], has been proposed to search for an optimal network architecture and reduce manual design efforts. Automating augmentation using NAS has recently been proposed in the field of data augmentation [29], [30], [32]. AutoAugment [29] uses an RNN controller to augment training data with a randomly selected augmentation method. Initially, all augmentation methods are explored uniformly, and we find optimal augmentation techniques that yield the best performance based on a reward function. Although AutoAugment has achieved promising results, it is less efficient and expensive. In addition, PBA [32] generates augmentation policies based on population-based training [33]. Fast AutoAugment [30] uses hyperparameter optimization to explore optimal augmentation policies. Unlike the above-mentioned methods, in this work, we focus on developing an automatic augmentation method for the field of FSL. We adopt the augmentation strategy in [30] to accelerate our augmentation phase, which will be described in the following section.

III. METHODOLOGY

We present an automatic augmentation strategy for the metric-based FSL without requiring domain knowledge and painful design efforts. The proposed efficient data augmentation network, termed EDANet, consists of augmentation and classification phases. In the augmentation phase, augmentation data are obtained by automatically exploring the most efficient augmentation policy for the FSL model from the original data. In the classification phase, the FSL network is trained with the augmented training data, and then the performance is evaluated with the original test dataset. We describe the proposed augmentation framework and few-shot augmentation with the proposed framework in Section III-A. In Section III-B, we apply EDANet to popular FSL problems.

A. EDANet: EFFICIENT DATA AUGMENTATION NETWORK

Manual data augmentation techniques generally require expert knowledge and painstaking design efforts despite the advantage of improving performance by enriching training data. As a remedy, we automate the augmentation procedure to find an optimal strategy and improve task performance. The search space of the automatic augmentation process contains diverse augmentation techniques, listed in Table 1. An augmentation operation $\mathcal{O}$ receives a sample x given the magnitude λ. The result can be either $\mathcal{O}(x;\lambda)$ with probability p or the original x itself with probability $1-p$. For example, when the rotation is selected as an augmentation operation, the magnitude becomes the degree. In the augmentation phase, we first introduce the search space of the augmentation techniques (operations) and perform density

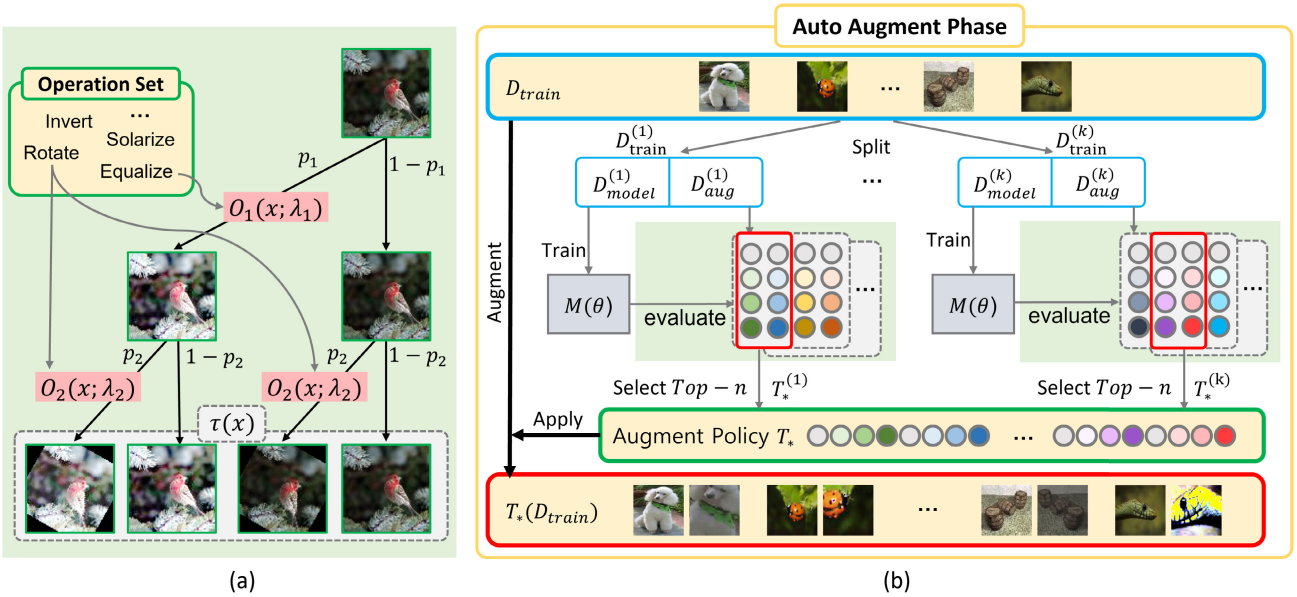

FIGURE 2. An overall procedure of automatic augmentation by EDANet. (a) An example of augmentation from the searched policies. Each operation O_i given the magnitude λ_i is called with the probability p_i. (b) The proposed method splits the training data set $\mathcal{D}_{train}$ into k-folds, each of which consists of two data sets $\mathcal{D}_{model}$ and $\mathcal{D}_{aug}$. The model parameter θ is trained on each $\mathcal{D}_{model}$. After training model $M(\theta)$, the algorithm evaluates the augmentation policies on $\mathcal{D}_{aug}$. The top n policies (e.g., eight (in the above figure)) obtained from each of the k-folds are appended to an augmentation list $\mathcal{T}_*(\mathcal{D}_{train})$. Finally, we measure the performance of the network model with $\mathcal{T}_*(\mathcal{D}_{train})$.

matching to find optimal augmentation policies. Figure 2 (b) shows the augmentation phase of EDANet. We follow the search strategy presented in [30] to find a good augmentation policy for FSL.

TABLE 1. Data augmentation include geometric transformations, color transformations, etc. Affine transformation is a type of geometric transformation, such as rotation and translation. Color enhancement performs augmentation in the color channel space, such as invert and equalize. Recently, other image manipulation methods have been proposed, such as cutout [13] and sample pairing [15].

	Operation	Magnitude λ
Affine transformation	ShearX	Continuous
	ShearY	Continuous
	TranslateX	Continuous
	TranslateY	Continuous
	Rotate	Continuous
Color enhancement	AutoContrast	None
	Invert	None
	Equalize	None
	Solarize	Discrete
	Posterize	Discrete
	Contrast	Continuous
	Color	Continuous
	Brightness	Continuous
	Sharpness	None
Others	Cutout	Discrete
	Sample Pairing	Continuous

1) AUTO AUGMENTATION PHASE

Let $\mathcal{O}$ denote the set of augmentation operations, which transforms an input sample by applying the operation. Specifically, each augmentation operation $\mathcal{O}_i$ contains a parameter, the magnitude λ_i. The input samples are augmented by a policy consisting of t different sub-policies τ's, and each $\mathcal{O}_i(x;\lambda)$ is applied to transform x with the probability p. Figure 2 (a) illustrates an example of augmenting samples by each sub-policy τ. We can generate $\mathcal{T}(\mathcal{D})$ indicating a set of augmented samples of dataset $\mathcal{D}$ transformed by all sub-policies $\tau \in \mathcal{T}$:

$$\mathcal{T}(\mathcal{D}) = \bigcup_{\tau\in\mathcal{T}} \{(\tau(x), y) : (x, y) \in \mathcal{D}\}. \tag{1}$$

Note that the data set $\mathcal{D}$ is divided into k-folds. Each fold consists of $\mathcal{D}_{model}$ and $\mathcal{D}_{aug}$. We find a promising policy giving the highest performance for $\mathcal{D}_{aug}$ based on model $M(\theta)$ that is trained by $\mathcal{D}_{model}$. Using the Bayesian optimization approach [30], the top-n augmentation methods were selected based on the minimum error rates of classifier θ when predicting data set $\mathcal{D}_{aug}$. All the best augmentation methods were merged from each fold.

2) DENSITY MATCHING

Our goal is to find an efficient augmentation policy that matches the density of $\mathcal{D}_{model}$ and $\mathcal{D}_{aug}$. We followed the practice in [30] that divides the original $\mathcal{D}_{train}$ into $\mathcal{D}_{model}$ and $\mathcal{D}_{aug}$ used for learning the model parameter θ and exploring the augmentation policy $\mathcal{T}$, respectively. To find a set of learned augmentation policies, we have the following

$$\mathcal{T}_* = \underset{\tau}{\operatorname{argmax}}\, \mathcal{R}(\theta \mid \mathcal{D}_{aug}), \tag{2}$$

where $\mathcal{R}(\theta \mid \mathcal{D}_{aug})$ gives the accuracy using the model parameter θ for $\mathcal{D}_{aug}$. We can find the policy based on the model trained with $\mathcal{D}_{model}$. In other words, it minimizes the distance between the density of $\mathcal{D}_{model}$ and the density of

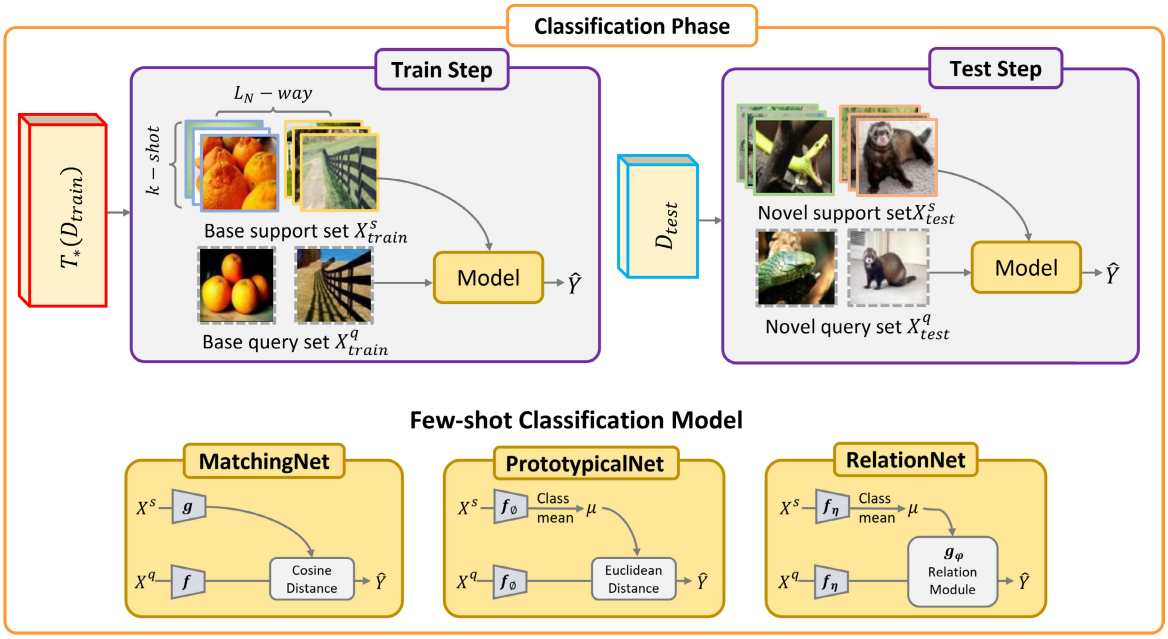

FIGURE 3. A graphical illustration of the classification phase. We search for an efficient augmentation policy in the automatic augmentation phase and perform episodic training for few-shot classification. The purple boxes are an example of an episode, and each episode consists of a support set and a query set. The yellow boxes represent FSL models and we train one of the models using the augmented dataset $\mathcal{T}_*(\mathcal{D}_{train})$. We test the model with $\mathcal{D}_{test}$.

$\mathcal{T}(\mathcal{D}_{aug})$. We optimize the search process of augmentation strategies in EDANet using Bayesian optimization [30].

3) CLASSIFICATION PHASE

After the augmentation phase, we learn few-shot classification networks with the augmented dataset $\mathcal{T}_*(\mathcal{D}_{train})$, which contains $\{(x_1, y_1), \ldots, (x_n, y_n)\}$, where x is a sample and y is the class label. Afterward, we test the proposed method with the original test set.

Figure 3 shows the classification phase of the proposed method. In the training stage, each task takes the form of L_N-way k-shot, and it consists of a support set X^s and a query set X^q. Then, we train the classification model M that minimizes the L_N-way prediction loss for the query set, also known as episodic training. In the testing stage, the episode consists of a novel support set and a novel query set. The classification model M can predict novel classes. The model M can use MatchingNet [19], PrototypicalNet [20], or RelationNet [21] as the backbone of the proposed method, which is described in the following section.

B. FEW-SHOT LEARNING OBJECTIVE FUNCTIONS

The support set and query set for training FSL are extracted from $\mathcal{D}_{aug}$ and $\mathcal{D}_{model}$, respectively. The support set X^s contains m examples as $(x^s_1, y^s_1), \cdots, (x^s_i, y^s_i), \cdots, (x^s_m, y^s_m)$. The query set X^q contains t examples as $(x^q_1, y^q_1), \cdots, (x^q_j, y^q_j), \cdots, (x^q_t, y^q_t)$.

1) MATCHING NETWORKS

MatchingNet [19], given a query sample x^q (a novel unseen sample), predicts the class of x^q by comparing it with the support set. The prediction of the class for the query sample is defined as follows:

$$\hat{y} = \sum_{i=1}^{k} Attention(x^q, x^s_i) \cdot y^s_i, \tag{3}$$

where $\hat{y}$ is the predicted class of the query sample. $Attention(\cdot, \cdot)$ is a simple attention mechanism between $\hat{x}$ and x^s_i, which is the softmax function over the cosine distance between $\hat{x}$ and x^s_i as follows:

$$Attention(\hat{x}, x^s_i) = \frac{e^{cos(f(\hat{x}), g(x^s_i))}}{\sum_{j=1}^{k} e^{cos(f(\hat{x}), g(x^s_j))}}. \tag{4}$$

Specifically, two embedding functions, f and g, are used to learn embeddings of $\hat{x}$ (input in the query set) and x^s_i (input in the support set), respectively. We convert the support set label into one-hot encoding and multiply them using the attention mechanism to obtain the probability of $\hat{y}$ belonging to each class in the support set. Then, we select $\hat{y}$ with the maximum probability value as the class label.

2) PROTOTYPICAL NETWORKS

PrototypicalNet [20] learns the embeddings of the data points using the embedding function, f_ϕ, where ϕ is the set of parameters of the embedding function. The prototype represents the mean embedding of data points in each class. The prototype c_j of the j-th class is calculated using the embedding of data points for the class as

$$c_j = \frac{1}{|X^s|} \sum_i f_\phi(x^s_i). \tag{5}$$

Then, the softmax function is applied after calculating the Euclidean distance $d(\cdot, \cdot)$ between the query set embedding

and class prototype. Through this, the probability of the query sample of the class j is predicted as follows:

$$p(y=j|x^q) = \frac{exp(-d(f_\phi(x^q), c_j))}{\sum_{j'} exp(-d(f_\phi(x^q), c_{j'}))}. \quad (6)$$

Finally, we define the loss function as the negative log probability of the class j as

$$\mathcal{L}(\phi) = -\log p(y=j|x^q), \quad (7)$$

and we minimize the loss using SGD.

3) RELATION NETWORKS

RelationNet [21] consists of two important functions: the embedding function, denoted by f_η, and the relation function, denoted by g_φ, where η and φ are the parameters of the embedding and relation functions, respectively. This network first takes a sample, x_i^s, from the support set and feeds it into the embedding function to extract the feature. Similarly, this method learns the embedding of a query sample x_j^q by passing through the embedding function, $f_\eta(\cdot)$. Then, it combines $f_\eta(x_i^s)$ and $f_\eta(x_j^q)$ using the concatenation operation, i.e., $concat(f_\eta(x_i^s), f_\eta(x_j^q))$, which is fed into the relation function, g_φ. The function generates a relation score ranging from 0 to 1:

$$r_{ij} = g_\varphi(concat(f_\eta(x_i^s), f_\eta(x_j^q))). \quad (8)$$

This represents the similarity between the samples in the support and query sets. For RelationNet, we used the mean squared error as the loss function, computed as follows:

$$\arg\min_{\varphi,\eta} \sum_{i=1}^{m}\sum_{j=1}^{n}(r_{ij}-\alpha)^2, \quad (9)$$

where α is 1 if $y_i^s == y_j^q$ and is 0 otherwise.

IV. EXPERIMENTS

In this section, we present the results of the proposed approach on the mini-ImageNet and Omniglot datasets and compare the results with other few-shot approaches. We summarize the proposed method as follows. The proposed method first finds an efficient augmentation policy in the augmentation phase and applies it to the original training set $\mathcal{D}_{train}$ to provide $\mathcal{T}_*(\mathcal{D}_{train})$. Then, we added the augmented data to the $\mathcal{D}_{train}$ and used it in the classification phase. In the classification phase, the few-shot models described in Section III-B are trained with the augmented dataset $\mathcal{T}_*(\mathcal{D}_{train})$ and evaluated with the test set $\mathcal{D}_{test}$. We ran every experimental scenario independently five times, reporting the average results. All experiments were implemented based on the PyTorch library [36].

A. SETTINGS

1) DATASETS

We used the mini-ImageNet and Omniglot benchmarks to evaluate the few-shot classification algorithms. The mini-ImageNet dataset proposed in [19] was derived from the ILSVRC-12 dataset [11]. It consists of 60,000 color images of size 84×84 and has 100 classes with 600 examples each. We split the dataset into 64 training, 16 validation, and 20 test classes. Omniglot [18] contains 1,623 handwritten characters from 50 different alphabets. There are 20 examples associated with each character, and each example is drawn by different people. We applied automatic augmentation to 1,200 characters for training and used the remaining 423 characters for testing. We followed a similar procedure for the metric-based approaches by resizing the image into 28×28. Metric-based approaches [19]–[21] usually augment the training set by rotating the images. However, for the mini-ImageNet dataset, we used the augmentation techniques shown in Table 1 to find and apply the optimal augmentation policy. Note here that since the Omniglot data set is a grayscale image, of the 16 operations listed in Table 1, only five affine transformation methods were chosen as the augmentation candidates. The data set to which the efficient augmentation methods are applied is added to the original training set through the augmentation phase. Afterward, a few-shot classification task is performed through the classification phase. We compared this method with the augmentation-based approaches, SGM [22], PMN [23], and IDeMeNet [24], for few-shot recognition.

2) IMPLEMENTATION DETAILS

Following the practice in [30], we applied the steps to make implementation easier. First, we split the training data, $\mathcal{D}_{train}$, into five folds. Second, we performed exploration-and-exploitation using HyperOpt [35] to search for the optimal augmentation policies. We selected augmentation operations from the PIL Python library.[1] We applied test-time-augmentation [34] to estimate the appropriate augmentation policy without repeated training. After training the FSL model, an effective augmentation policy is estimated from a pool of candidate augmentation policies. Finally, the augmentation policy that achieves the highest performance is selected. The original implementations of MatchingNet [19], PrototypicalNet [20], and RelationNet [21] use the embedding architecture containing four convolution layers (Conv-4). Other than the architecture, we used ResNet-18 and ResNet-50 as additional embedding functions for diverse experiments. We employed the same training procedure for the embedding functions as existing approaches. We used the 5-way classification setting in the experiments.

B. RESULTS

1) MINI-ImageNet

We performed the experiments on 5-way 1-shot and 5-shot according to the common setup in metric-based FSL. Since the mini-ImageNet dataset consists of RGB images, EDANet considered all 16 augmentation operations listed in Table 1. We obtained the classification accuracy by averaging over 600 randomly generated episodes from the test set. We compared EDANet with the existing augmentation-based algorithms SGM [22], PMN [23], and IDeMeNet [24].

[1] https://python-pillow.org/

TABLE 2. Few-shot classification results on mini-ImageNet.

Method	Model	Backbone	Distance Metric	5-Way 1-shot	5-Way 5-shot
SGM [22]	MatchingNet	ResNet-50	Cosine Dist.	45.1%	72.7%
PMN [23]	PrototypicalNet	ResNet-50	Euclidean Dist.	57.6%	71.9%
IDeMeNet [24]	PrototypicalNet	ResNet-50	Euclidean Dist.	59.14%	74.63%
No Augmentation	MatchingNet	Conv-4	Cosine Dist.	49.37%	63.87%
	PrototypicalNet	Conv-4	Euclidean Dist.	57.01%	75.11%
	RelationNet	Conv-4	Relation Score	57.84%	74.61%
EDANet	MatchingNet	ResNet-50	Cosine Dist.	50.36%	67.8%
	PrototypicalNet	ResNet-50	Euclidean Dist.	63.35%	79.74%
	RelationNet	ResNet-50	Relation Score	63.51%	75.03%

TABLE 3. Few-shot classification results on Omniglot.

Method	Model	Backbone	Distance Metric	5-Way 1-shot	5-Way 5-shot
No Augmentation (our impl.)	MatchingNet	Conv-4	Cosine Dist.	91.81%	92.33%
	PrototypicalNet	Conv-4	Euclidean Dist.	92.29%	93.22%
	RelationNet	Conv-4	Relation Score	92.41%	93.50%
Manual Augmentation (our impl.)	MatchingNet	Conv-4	Cosine Dist.	93.25%	94.51%
	PrototypicalNet	Conv-4	Euclidean Dist.	95.01%	95.88%
	RelationNet	Conv-4	Relation Score	95.36%	96.02%
Auto Augmentation (EDANet)	MatchingNet	Conv-4	Cosine Dist.	94.67%	95.35%
	PrototypicalNet	Conv-4	Euclidean Dist.	96.12%	96.78%
	RelationNet	Conv-4	Relation Score	96.64%	97.13%
No Augmentation (our impl.)	MatchingNet	ResNet-18	Cosine Dist.	94.18%	95.13%
	PrototypicalNet	ResNet-18	Euclidean Dist.	94.87%	95.88%
	RelationNet	ResNet-18	Relation Score	95.51%	96.21%
Manual Augmentation (our impl.)	MatchingNet	ResNet-18	Cosine Dist.	96.55%	97.81%
	PrototypicalNet	ResNet-18	Euclidean Dist.	97.58%	98.27%
	RelationNet	ResNet-18	Relation Score	97.91%	98.31%
Auto Augmentation (EDANet)	MatchingNet	ResNet-18	Cosine Dist.	98.11%	98.73%
	PrototypicalNet	ResNet-18	Euclidean Dist.	98.89%	99.62%
	RelationNet	ResNet-18	Relation Score	99.13%	99.57%

Moreover, EDANet was trained under three different embedding functions (models) described in Section III-B.

We report the accuracy of the compared methods on mini-ImageNet in Table 2. In addition, EDANet achieves the best accuracy when using PrototypicalNet as the embedding function. We employed ResNet-50 as another backbone for a fair comparison with data augmentation-based approaches [22]–[24], which performs better than the Conv-4 architecture. The deeper backbone network performs better than shallow networks on every shot. Notably, the proposed approach based on Conv-4 gives similar or better performance than the compared methods employing the ResNet-50 backbone. This shows that the automatic augmentation method is effective in FSL, compared to other approaches with hand-crafted augmentation rules. Further, 5-shot learning achieves higher accuracy than 1-shot learning, and the augmented dataset helps the network generalize better. Compared to SGM [22], PMN [23], and IDeMeNet [24], we observe that EDANet performs better than those approaches in both 1-shot and 5-shot. The 1-shot performance of the proposed method improved over the compared approaches. For EDANet based on MatchingNet, the 5-shot learning performance is lower than that of the SGM. However, EDANet performs better than the other methods on average for 5-shot learning.

TABLE 4. Few-shot classification results for ablation study on mini-ImageNet.

Method	Model	Distance Metric	Backbone	5-Way 1-shot	5-Way 5-shot
No Augmentation (Baseline)	MatchingNet [19]	Cosine Dist.	Conv-4	43.56%	55.31%
	PrototypicalNet [20]	Euclidean Dist.	Conv-4	49.42%	68.20%
	RelationNet [21]	Relation Score	Conv-4	50.44%	65.32%
Manual Augmentation	MatchingNet	Cosine Dist.	Conv-4	45.60%	58.95%
	PrototypicalNet	Euclidean Dist.	Conv-4	52.63%	70.33%
	RelationNet	Relation Score	Conv-4	53.22%	68.12%
Auto Augmentation (EDANet)	MatchingNet	Cosine Dist.	Conv-4	49.37%	63.87%
	PrototypicalNet	Euclidean Dist.	Conv-4	57.01%	75.11%
	RelationNet	Relation Score	Conv-4	57.84%	74.61%

2) OMNIGLOT

We conducted another experiment on Omniglot and computed the few-shot classification accuracy by averaging over 1,000 randomly generated episodes from the test set. We experimented with the proposal based on two different backbone architectures in metric-based few-shot classification. For comparison, we compared the proposed method with MatchingNet [19], PrototypicalNet [20], and RelationNet [21] without data augmentation and with manual augmentations (rotation, crop, and flip). Since the Omniglot dataset contains black-and-white binary images, we applied a total of five affine transformation methods except for the color enhancement techniques in Table 1.

The results of the compared approaches are listed in Table 3. The proposed method achieved better performance than the baseline methods for the dataset when PrototypicalNet was used under 5-way 1-shot learning and RelationNet was used under 5-way 5-shot learning. Compared with manual augmentation using the Conv-4 backbone, the proposed method achieves better performance by about 1%. As presented in the table, there is a performance gap depending on the distance metric, and RelationNet using the relation score produces the best performance among the distance metrics for both 1-shot and 5-shot. PrototypicalNet and RelationNet outperform MatchingNet using the cosine distance, and their performance difference is marginal. Further, EDANet using the automatic augmentation strategy performs better than the manual augmentation-based approaches under the same models (embedding function). Even with a larger backbone network (ResNet-18), the method improves the performance by 2% to 3%. The experiments show that EDANet finds an optimal data augmentation strategy from the pool of candidate augmentation techniques for metric-based FSL.

C. ABLATION STUDY

1) EFFECTIVENESS OF AUTOMATIC AUGMENTATION

We demonstrate the effectiveness of the proposed EDANet that explores an optimal augmentation strategy from a pool of candidate strategies in metric-based FSL and compare it with manual augmentation approaches. Table 4 shows the results with respect to different augmentation strategies. No augmentation follows the method suggested in existing papers without adding augmentation techniques. Manual augmentation applies three commonly used augmentation techniques; flipping, cropping, and rotating. Auto augmentation corresponds to EDANet, which explores the best augmentation rule among 16 augmentations techniques listed in Table 1. In the study, we used the small backbone network (Conv-4). The proposed automatic augmentation yields the best accuracy compared to other methods by large margins. The proposed method automatically explores optimal augmentation methods without requiring expert effort, resulting in more than 7% performance improvement over no augmentation for both 1-shot and 5-shot results. Similarly, this method outperforms the manual augmentation method by a margin of 4% to 6%. The experiments show that EDANet is a promising candidate to achieve competitive performance without laborious manual design costs.

2) RESULTS OF VARIOUS AUGMENTATION METHODS

We conducted additional experiments to observe the performance of some augmentation operations in Table 1. We selected eight augmentation operations (including no augmentation) and applied each of them as an augmentation method under the MatchingNet framework. The FSL results from the augmentation strategy are summarized in Table 5. The performance of MatchingNet without augmentation gives the lowest performance (37.1% for 1-shot learning and 50.4% for 5-shot learning). All augmentation operations give a performance improvement of 2% or more over the method without augmentation. We also observed that augmentation methods (e.g., translate, contrast, brightness, etc.) other than rotation improve the task performance in metric-based FSL. The color enhancement operations (contrast, brightness, saturation, and hue) give a 2.9% higher performance improvement on average than the affine transformation operations (rotate, translateY, and translateX).

TABLE 5. Performance with respect to different usages of augmentation methods using MatchingNet on mini-ImageNet. A slight gap exists between the results in [19] and those in the proposed implementation due to the differences in implementation.

Model	Augmentation Method	5-Way 1-shot	5-Way 5-shot
MatchingNet [19]	Rotate (90, 180, 270 degrees)	41.2%	56.2%
MatchingNet (our impl.)	No augmentation	37.1%	50.4%
	Rotate (90, 180, 270 degrees)	40.2%	55.6%
	Rotate (1 $\sim$ 270 degrees)	40.4%	55.4%
	TranslateX	41.5%	57.2%
	TranslateY	41.4%	56.7%
	Contrast	43.5%	59.0%
	Brightness	44.3%	59.6%
	Saturation	43.8%	58.8%
	Equalize	43.9%	59.3%
	Rotate + Contrast	46.5%	60.7%
	TranslateX + Equalize	46.8%	61.2%
	Rotate + Brightness	47.1%	61.5%

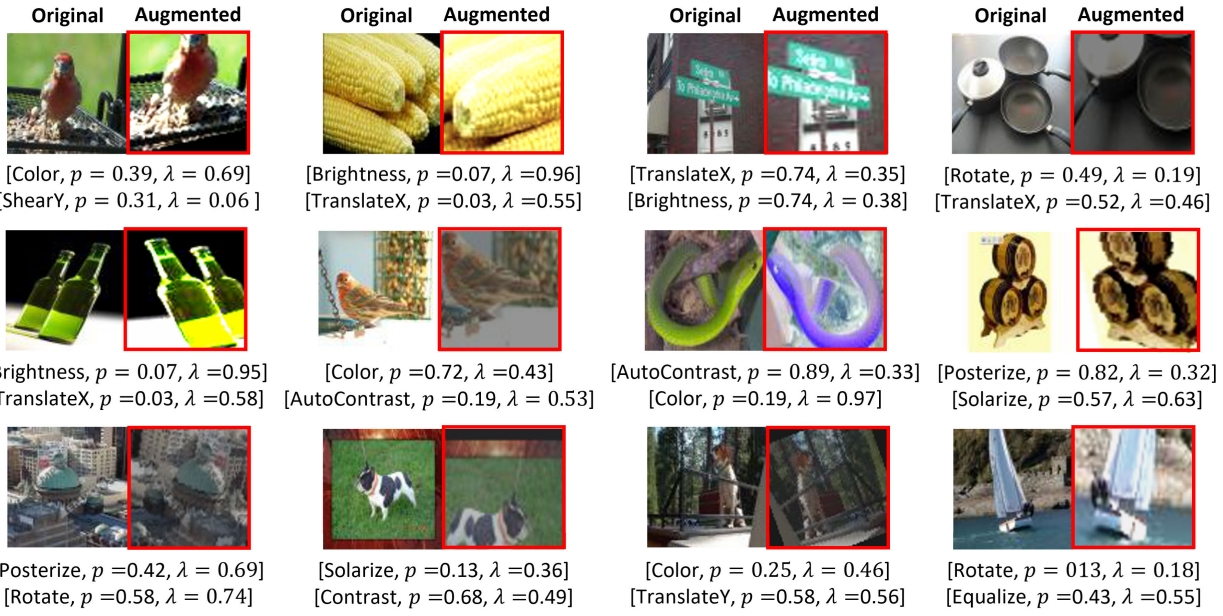

FIGURE 4. Examples of augmented images and their corresponding policies and parameter values in the proposed method on mini-ImageNet.

In addition, we applied combinations of the augmentation methods to see whether the combination leads to performance improvement. The number of combinations of choosing two out of the above eight selected augmentation operations is 28. We randomly selected three of these methods. The table shows that combining two augmentation methods performs better than the standalone augmentation method (at least 2% improvement). Here, we note that since there are a large number of combinations to select two of the 16 augmentation operations, it is nearly impossible to select appropriate augmentation methods manually. Due to this reason, we can say that our EDANet can be a promising strategy.

Figure 4 shows some examples of augmented images with their corresponding policies and parameter values selected in EDANet. In addition, EDANet provides the probability and magnitude for each policy combination and selects an augmented image from the combination of multiple augmentation operations. This outcome reveals that the proposed method is capable of exploring various augmentation policies for different images.

V. CONCLUSION

Data augmentation is one of the promising methods for the success of FSL that requires few data. In this work, we have proposed an efficient automatic augmentation approach for FSL to explore the best augmentation strategies from a pool of candidate augmentation operations and reduce hand-crafted design efforts. The proposed method, EDANet, searches for

a combination of augmentation techniques given various candidates augmentation methods using Bayesian optimization and density matching. We have applied the proposed method to three popular FSL baseline models using different distance metrics. The experimental results have shown that the automatic augmentation rule yields better performance than manual augmentation-based counterparts under the same model, showing its effectiveness in FSL.

REFERENCES

[1] A. Krizhevsky, I. Sutskever, and G. E. Hinton, "Imagenet classification with deep convolutional neural networks," in *Proc. Adv. Neural Inf. Process. Syst.*, vol. 25, 2012, pp. 1097–1105.
[2] 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.
[3] M. Tan and Q. Le, "Efficientnet: Rethinking model scaling for convolutional neural networks," in *Proc. Int. Conf. Mach. Learn.*, Apr. 2019, pp. 6105–6114.
[4] J. Redmon, S. Divvala, R. Girshick, and A. Farhadi, "You only look once: Unified, real-time object detection," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2016, pp. 779–788.
[5] T.-Y. Lin, P. Goyal, R. Girshick, K. He, and P. Dollár, "Focal loss for dense object detection," in *Proc. IEEE Int. Conf. Comput. Vis. (ICCV)*, Oct. 2017, pp. 2980–2988.
[6] L. C. Chen, G. Papandreou, and I. Kokkinos, "DeepLab: Semantic image segmentation with deep convolutional nets, atrous convolution, and fully connected CRFs," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 40, no. 4, pp. 834–848, Jun. 2016.
[7] K. He, G. Gkioxari, P. Dollár, and R. Girshick, "MASK R-CNN," in *Proc. IEEE Int. Conf. Comput. Vis.*, Oct. 2017, pp. 2961–2969.
[8] B. Zoph and Q. V. Le, "Neural architecture search with reinforcement learning," 2016, *arXiv:1611.01578*.
[9] 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.
[10] A. Krizhevsky and G. Hinton, "Learning multiple layers of features from tiny images," Univ. Toronto, Tech. Rep., 2009.
[11] O. Russakovsky, J. Deng, H. Su, J. Krause, S. Satheesh, S. Ma, Z. Huang, A. Karpathy, A. Khosla, M. Bernstein, A. C. Berg, and L. Fei-Fei, "ImageNet large scale visual recognition challenge," *Int. J. Comput. Vis.*, vol. 115, no. 3, pp. 211–252, Dec. 2015.
[12] H. Zhang, M. Cisse, Y. N. Dauphin, and D. Lopez-Paz, "Mixup: Beyond empirical risk minimization," 2017, *arXiv:1710.09412*.
[13] T. DeVries and G. W. Taylor, "Improved regularization of convolutional neural networks with cutout," 2017, *arXiv:1708.04552*.
[14] S. Yun, D. Han, S. Chun, S. J. Oh, Y. Yoo, and J. Choe, "CutMix: Regularization strategy to train strong classifiers with localizable features," in *Proc. IEEE/CVF Int. Conf. Comput. Vis. (ICCV)*, Oct. 2019, pp. 6023–6032.
[15] H. Inoue, "Data augmentation by pairing samples for images classification," 2018, *arXiv:1801.02929*.
[16] C. Shorten and T. M. Khoshgoftaar, "A survey on image data augmentation for deep learning," *J. Big Data*, vol. 6, no. 1, pp. 1–48, Dec. 2019.
[17] 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.
[18] B. Lake, R. Salakhutdinov, J. Gross, and J. Tenenbaum, "One shot learning of simple visual concepts," in *Proc. Annu. Meeting Cognit. Sci. Soc.*, vol. 33, 2011, pp. 1–7.
[19] O. Vinyals, C. Blundell, T. Lillicrap, and D. Wierstra, "Matching networks for one shot learning," in *Proc. Adv. Neural Inf. Process. Syst.*, vol. 29, 2016, pp. 3630–3638.
[20] J. Snell, K. Swersky, and R. S. Zemel, "Prototypical networks for few-shot learning," 2017, *arXiv:1703.05175*.
[21] 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.
[22] 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. 3018–3027.
[23] 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.
[24] 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.
[25] C. Finn, P. Abbeel, and S. Levine, "Model-agnostic meta-learning for fast adaptation of deep networks," in *Proc. Int. Conf. Mach. Learn.*, Jul. 2017, pp. 1126–1135.
[26] Z. Li, F. Zhou, F. Chen, and H. Li, "Meta-SGD: Learning to learn quickly for few-shot learning," 2017, *arXiv:1707.09835*.
[27] E. Schonfeld, S. Ebrahimi, S. Sinha, T. Darrell, and Z. Akata, "Generalized zero-and few-shot learning via aligned variational autoencoders," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit.*, Jun. 2019, pp. 8247–8255.
[28] A. Li, T. Luo, Z. Lu, T. Xiang, and L. Wang, "Large-scale few-shot learning: Knowledge transfer with class hierarchy," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2019, pp. 7212–7220.
[29] E. D. Cubuk, B. Zoph, D. Mane, V. Vasudevan, and Q. V. Le, "AutoAugment: Learning augmentation strategies from data," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2019, pp. 113–123.
[30] S. Lim, I. K. T. Kim, C. Kim, and S. Kim, "Fast autoaugment," in *Proc. Adv. Neural Inf. Process. Syst.*, vol. 32, 2019, pp. 6665–6675.
[31] R. Hataya, J. Zdenek, K. Yoshizoe, and H. Nakayama, "Faster autoaugment: Learning augmentation strategies using backpropagation," in *Proc. Eur. Conf. Comput. Vis.* Cham, Switzerland: Springer, Aug. 2020, pp. 1–16.
[32] D. Ho, E. Liang, X. Chen, I. Stoica, and P. Abbeel, "Population based augmentation: Efficient learning of augmentation policy schedules," in *Proc. Int. Conf. Mach. Learn.*, May 2019, pp. 2731–2741.
[33] M. Jaderberg, V. Dalibard, S. Osindero, W. M. Czarnecki, J. Donahue, A. Razavi, O. Vinyals, T. Green, I. Dunning, K. Simonyan, C. Fernando, and K. Kavukcuoglu, "Population based training of neural networks," 2017, *arXiv:1711.09846*.
[34] I. Kim, Y. Kim, and S. Kim, "Learning loss for test-time augmentation," 2020, *arXiv:2010.11422*.
[35] J. Bergstra, D. Yamins, and D. D. Cox, "Making a science of model search: Hyperparameter optimization in hundreds of dimensions for vision architectures," in *Proc. 30th Int. Conf. Mach. Learn. (ICML)*, Jun. 2013, p. 115.
[36] A. Paszke, S. Gross, F. Massa, A. Lerer, J. Bradbury, G. Chanan, and S. Chintala, "Pytorch: An imperative style, high-performance deep learning library," in *Proc. Adv. Neural Inf. Process. Syst.*, vol. 32, 2019, pp. 8026–8037.

WONHEE CHO received the B.S. degree in applied statistics from Chung-Ang University, Seoul, South Korea, in 2019, where she is currently pursuing the M.S. degree with the School of Computer Science and Engineering. Her research interests include deep learning, machine learning, few-shot learning, and computer vision.

EUNWOO KIM (Member, IEEE) received the B.S. degree in electrical and electronics engineering from Chung-Ang University, Seoul, South Korea, in 2011, and the M.S. and Ph.D. degrees in electrical engineering and computer science from Seoul National University, Seoul, in 2013 and 2017, respectively. From 2017 to 2018, he was a Postdoctoral Researcher with the Department of Electrical Engineering and Computer Science, Seoul National University. From 2018 to 2019, he was a Postdoctoral Researcher with the Department of Engineering Science, University of Oxford, Oxford, U.K. He is currently an Assistant Professor with the School of Computer Science and Engineering, Chung-Ang University. His research interests include machine learning, deep learning, model optimization, robotics, and computer vision.