Neural Networks 138 (2021) 140–149

Contents lists available at ScienceDirect

Neural Networks

journal homepage: www.elsevier.com/locate/neunet

# Self-augmentation: Generalizing deep networks to unseen classes for few-shot learning

Jin-Woo Seo [a,1], Hong-Gyu Jung [a,1], Seong-Whan Lee [b,*]

[a] Department of Brain and Cognitive Engineering, Korea University, Anam-dong, Seongbuk-gu, Seoul, 02841, Republic of Korea
[b] Department of Artificial Intelligence, Korea University, Anam-dong, Seongbuk-gu, Seoul, 02841, Republic of Korea

ARTICLE INFO

*Article history:*
Received 30 July 2020
Received in revised form 8 January 2021
Accepted 8 February 2021
Available online 17 February 2021

*Keywords:*
Few-shot learning
Classification
Generalization
Knowledge distillation

ABSTRACT

Few-shot learning aims to classify unseen classes with a few training examples. While recent works have shown that standard mini-batch training with carefully designed training strategies can improve generalization ability for unseen classes, well-known problems in deep networks such as memorizing training statistics have been less explored for few-shot learning. To tackle this issue, we propose self-augmentation that consolidates self-mix and self-distillation. Specifically, we propose a regional dropout technique called self-mix, in which a patch of an image is substituted into other values in the same image. With this dropout effect, we show that the generalization ability of deep networks can be improved as it prevents us from learning specific structures of a dataset. Then, we employ a backbone network that has auxiliary branches with its own classifier to enforce knowledge sharing. This sharing of knowledge forces each branch to learn diverse optimal points during training. Additionally, we present a local representation learner to further exploit a few training examples of unseen classes by generating fake queries and novel weights. Experimental results show that the proposed method outperforms the state-of-the-art methods for prevalent few-shot benchmarks and improves the generalization ability.



## 1. Introduction

Deep networks have achieved remarkable performance in recognition problems (He, Zhang, Ren, & Sun, 2016; Szegedy et al., 2015; Zhu, Liu, Zhu, Deng, & Sun, 2020) over hand-crafted features (Kang, Han, Jain, & Lee, 2014; Roh, Kim, Park, & Lee, 2007; Roh, Shin, & Lee, 2010). Assuming a large-scale training dataset is available, most studies focus on training deep networks on base classes to test unseen *images* of trained classes. However, there is a growing interest in mimicking human abilities such as generalizing a recognition system to classify *classes* that have never been seen before (Bulthoff, Lee, Poggio, & Wallraven, 2003; Liu, Gao, Gao, Han, & Shao, 2020; Vinyals, Blundell, Lillicrap, Wierstra, et al., 2016). In particular, few-shot learning assumes only a few training examples are available for unseen classes. This is challenging since it is highly possible that a few training examples will lead to network overfitting.

One paradigm for this challenge is meta-learning (Finn, Abbeel, & Levine, 2017; Snell, Swersky, & Zemel, 2017; Vinyals et al., 2016), where a large-scale training set for base classes is divided into several subsets (typically called tasks) and the network learns how to adapt to those tasks. In each task, only a few training examples are given for each class to mimic the environment of a test set for unseen classes.

Meanwhile, recent works have shown that a network trained with standard supervision can produce reasonable performance on unseen classes (Dvornik, Schmid, & Mairal, 2019; Gidaris, Bursuc, Komodakis, Pérez, & Cord, 2019; Lifchitz, Avrithis, Picard, & Bursuc, 2019). In the training phase, this paradigm trains a network using a mini-batch sampled from a large-scale training dataset. In the test phase, unseen classes with a few training examples are evaluated using the same network. Thus, the goal is to develop a network that is generalizable to unseen classes by fully utilizing the knowledge learned from base classes.

Both paradigms share commonalities in that they leverage a large annotated collection. However, the following notable difference exists: Meta-learning learns to adapt quickly to new tasks by splitting base classes into several different tasks, whereas the standard supervision constructs a parameter space in which the *unseen classes* can be identified using only the information for classifying the base classes at once. While the latter paradigm is closely related to classifying the *unseen images* belonging to the base classes, only a few studies have taken advantage of lessons learned from the classical classification problem (Dvornik et al., 2019; Gidaris et al., 2019; Lee & Kim, 1999; Lifchitz et al., 2019).

* Corresponding author.
*E-mail address:* sw.lee@korea.ac.kr (S.-W. Lee).
[1] Equal contribution.

https://doi.org/10.1016/j.neunet.2021.02.007
0893-6080/

To tackle this issue, we take a closer look at the generalization ability of deep networks for few-shot learning. It is known that deep networks tend to have almost zero-entropy distributions as the softmax output produces one peaked value for a class (Verma et al., 2018). This overconfidence can occur even with randomly labeled training data as deep networks are likely to just memorize the training statistics (Thulasidasan, Chennupati, Bilmes, Bhattacharya, & Michalak, 2019). In our problem setting, this memorization property directly affects the performance on unseen classes as we rely heavily on the network ability trained on the dataset of base classes. The problem even worsens as we cannot apply a simple transfer learning strategy given that we have only a few training examples for unseen classes. Thus, to overcome the memorization issues, it is important to induce uncertainty in predictions about input images and regularize the posterior probability (DeVries & Taylor, 2017; Pereyra, Tucker, Chorowski, Kaiser, & Hinton, 2017; Yun et al., 2019; Zhang, Xiang, Hospedales and Lu, 2018).

With this in mind, we propose self-augmentation that incorporates regional dropout and knowledge distillation to improve the generalization ability[2] for few-shot leaning. Here, we use the self-augmentation term as we use input and output resources of the network itself to augment the generalization ability. Specifically, as one of the data augmentation techniques, we employ regional dropout, which substitutes a patch of an input image into other values such as zeros (DeVries & Taylor, 2017), a patch of another image (Yun et al., 2019), and another patch of the input image. We call the last regional dropout "self-mix" as it exchanges different patches of the input image itself. With this dropout effect, the generalization ability is improved as it prevents us from learning specific structures of a dataset. Furthermore, we found that an explicit regularization for the posterior probability is necessary to search for a proper manifold for unseen classes. To be specific, we utilize a backbone network that has auxiliary branches with its own classifier to enforce knowledge sharing. This sharing of knowledge forces each branch not to be over-confident in its predictions, thus improving the generalization ability. Cooperating with regional dropout, the experimental results show that knowledge distillation significantly boosts the performance on unseen classes.

Lastly, we present a fine-tuning method to exploit a few training examples given for unseen classes. As we train a network on base classes, we have the opportunity to improve the discriminative ability of the network for unseen classes using only 1 or 5 training examples.

To sum up, our main contributions are as follows:

(1) We present self-augmentation as a training framework to improve the generalization ability of deep networks. Specifically, we design consolidating regional dropout and knowledge distillation, which are less explored in the few-shot learning area.
(2) We show that the newly proposed regional dropout, called self-mix, produces state-of-the art results when cooperating with knowledge distillation.
(3) Lastly, we present a novel fine-tuning method, called a local representation learner, to exploit a few training examples of unseen classes, and show that the method improves the performance for all the few-shot learning benchmarks.

[2] Henceforth, we denote the term "generalization" as the ability to adapt to unseen classes, given a network trained on base classes.

## 2. Related work

### 2.1. Few-shot learning

The literature on few-shot learning considers training and test datasets that are disjoint in terms of classes. Depending on how the training set is handled, we can categorize it into two main branches: meta-learning and standard supervised learning.

Meta-learning approaches train a network by explicitly emulating the test environment for few-shot learning. Using a training dataset, $n$ classes are randomly chosen with $k$ training examples, and $T$ queries are also randomly picked. Then, learnable parameters are obtained from the $n \cdot k$ training examples, and a loss is generated using the $T$ queries. A network is learned to reduce the loss by repeating this task several times. As a result, meta-learning warms a network up to classify unseen classes with a few examples. Three approaches exist for this paradigm: (1) Metric-learning to reduce the distance among features of different classes (Li, Eigen, Dodge, Zeiler, & Wang, 2019; Oreshkin, López, & Lacoste, 2018; Snell et al., 2017; Sung et al., 2018; Vinyals et al., 2016; Ye, Hu, Zhan, & Sha, 2020; Zhang, Cai, Lin, & Shen, 2020), (2) optimization-based approaches to initialize a parameter space so that a few training examples of unseen classes can be quickly trained with the cross-entropy loss (Finn et al., 2017; Rusu et al., 2019; Sun, Liu, Chua and Schiele, 2019), and (3) weight generation methods to directly generate classification weights used for unseen classes (Gidaris & Komodakis, 2018, 2019; Qi, Brown, & Lowe, 2018).

In contrast, the standard supervised learning approaches train a network as usual without splitting a training dataset into several tasks. In other words, this approach utilizes the training dataset as in the classical classification problem, but it aims to generalize unseen classes. To achieve this, dense classification applies the classification loss to all spatial information of an activation map to maximally exploit local information (Lifchitz et al., 2019). A previous study used self-supervision and showed that the auxiliary loss without labels can extract discriminative features for few-shot learning (Gidaris et al., 2019). An ensemble method using multiple networks was also proposed to resolve the high-variance issue in few-shot learning (Dvornik et al., 2019).

### 2.2. Generalization

Many efforts have been made to understand the generalization performance of deep learning (Guo, Pleiss, Sun, & Weinberger, 2017; Neyshabur, Bhojanapalli, McAllester, & Srebro, 2017; Pereyra et al., 2017; Thulasidasan et al., 2019; Ukita, 2020; Verma et al., 2018; Zhang, Bengio, Hardt, Recht, & Vinyals, 2017; Zhang, Xiang et al., 2018). Notably, it has been shown that deep networks easily adapt to random labels and are even well trained for images that appear as nonsense to humans (Zhang et al., 2017). Along the same lines, many works have found that deep networks produce overconfident classification predictions about an input, thus causing loss in the generalization performance (Müller, Kornblith, & Hinton, 2019; Pereyra et al., 2017; Thulasidasan et al., 2019; Zhang, Xiang et al., 2018). To resolve this issue, recently, regional dropout (DeVries & Taylor, 2017; Yun et al., 2019) and mixing up of two images (Tokozume, Ushiku, & Harada, 2018; Zhang, Cisse, Dauphin and Lopez-Paz, 2018) have been proposed as data augmentation techniques. Other researchers showed that label smoothing (Szegedy et al., 2015) and knowledge distillation (Hinton, Vinyals, & Dean, 2015; Lan, Zhu, & Gong, 2018; Sun, Yao, Zhou and Zhao, 2019; Zhang, Xiang et al., 2018) effectively mitigate the overfitting problem by regularizing the posterior probability. In this paper, we expand these findings and indicate that perturbing input and output information should

be extensively investigated for few-shot leaning. To this end, we propose a training framework that consolidates regional dropout and knowledge distillation, and further present a novel regional dropout called self-mix. In addition, we show that a novel fine-tuning method can be used to boost the performance of few-shot learning.

## 3. Methodology

### 3.1. General framework

In this paper, we are interested in training a network on base classes to be generalizable to unseen classes. Before elaborating on the proposed method, we introduce the general framework for training and inference.

#### 3.1.1. Training

We define a classifier as $C\left(f\left(\cdot;\Theta^{1:B}\right)\right)$, where $f(\cdot;\Theta^{1:B})$ is a feature extractor. Here, we denote the parameters from Block 1 to $B$ as $\Theta^{1:B}$, assuming that we use a block-wise network such as ResNet (He et al., 2016). For the classifier, we use the cosine similarity that has been exploited for few-shot learning (Gidaris & Komodakis, 2018; Qi et al., 2018). Thus, the $k$th output of the classifier for a training example $x_i$ can be defined as

$$C_k\left(f\left(x_i;\Theta^{1:B}\right)\right)=\text{softmax}\left(\tau\bar{f}_i^T\overline{w}_k\right), \tag{1}$$

where $\bar{f}_i$ is the L2 normalized feature for $x_i$ and $\overline{w}_k$ is the L2 normalized weight for the $k$th class. $\tau$ is used as a scale parameter for stabilized training (Chen, Liu, Kira, Wang, & Huang, 2019; Gidaris & Komodakis, 2018). Based on the definition, $C^{Base}$ is denoted as the classifier using base weights, and similarly $C^{Novel}$ is denoted using novel weights.

Then, we consider the mini-batch training with $N_{bs}$ examples and the cost function for our training method is expressed as

$$J(\Theta)=\frac{1}{N_{bs}}\sum_{i=1}^{N_{bs}}\ell\left(C\left(f\left(\tilde{x}_i;\Theta^{1:B}\right)\right);\tilde{y}_i\right)+R, \tag{2}$$

where there exist three components: (a) the virtual training example $\tilde{x}_i$ and label $\tilde{y}_i$, (b) a loss function $\ell$ and (c) a regularizer $R$. We sequentially elaborate on the components in the following subsections.

#### 3.1.2. Inference

After training base classes, we report the classification performance on unseen classes that have only a few training examples. We consider that a test dataset has $C^N$ classes, which are disjoint to $C^B$ classes for a training dataset. Thus, this inference process measures how well the network trained on base classes is generalized to unseen classes. For this measurement, we randomly sample $n$ classes from $C^N$ classes, and pick $k$ examples from each class. The typical numbers for few-shot learning are $n=5$ and $k=1$ or 5. This setting is called $n$-way $k$-shot classification.

After the sampling process, we generate the weight of the $j$th unseen class as follows:

$$w_j^N=\frac{1}{k}\sum_{i=1}^{k}f_{i,j}, \tag{3}$$

where $f_{i,j}$ is the feature of the $i$th example given for the $j$th unseen class. Then, a query $x_q$ is classified as

$$\underset{k}{\text{argmax}}\ C_k^{Novel}\left(f\left(x_q;\Theta^{1:B}\right)\right), \tag{4}$$

where $C_k^{Novel}$ is defined in Eq. (1) with the above novel weights. We iterate these sampling and inference processes several times to obtain the 95% confidence interval.

### 3.2. Self-augmentation

To improve the generalization performance, we propose a training framework called self-augmentation, which consolidates self-mix and knowledge distillation. Self-mix randomly picks a region of an input image and substitutes the pixels of the region into other values of the same image. We incorporate the self-mix in the few-shot learning problem and show that the generalization performance can be significantly boosted when collaborating with knowledge distillation. The overall architecture of the proposed method is shown in Fig. 1.

#### 3.2.1. Self-mix

Self-mix is applied to a raw input image to produce a transformed virtual example as follows:

$$\tilde{x}=T\left(x_i\right),$$

where $T: x_i[P_1]\rightarrow x_i[P_2]$ denotes the abuse of notation, the patch $P_1$ of $x_i$ is replaced by the patch $P_2$ of $x_i$. To be specific, we firstly sample a cropping region $P_1=(r_{a_1},r_{b_1},r_w,r_h)$ from an image. The x–y coordinates $(r_{a_1},r_{b_1})$ are sampled randomly and $(r_w,r_h)$ is set to a predefined size. If the patch exceeded the image boundary, we crop it. Then, a patch $P_2$ is sampled with the fixed $(r_w,r_h)$ and $(r_{a_2},r_{b_2})\left(\neq(r_{a_1},r_{b_1})\right)$ randomly chosen by ensuring not exceeding the image boundary.

#### 3.2.2. Self-distillation

Knowledge distillation has been studied to mitigate the overfitting problem by regularizing the posterior probability (Hinton et al., 2015; Lan et al., 2018; Sun, Yao et al., 2019; Zhang, Xiang et al., 2018). Although a recent work showed that knowledge distillation among multiple networks can ease off the high-variance characteristic in few-shot learning (Dvornik et al., 2019), this method requires 20 networks for the best performance. Thus, we incorporate self-distillation into our training framework, which employs auxiliary classifiers (Lan et al., 2018; Sun, Yao et al., 2019). The concept is to create independent predictions for an input image and share the information that has been learned by each classifier. To ensure that the auxiliary classifiers share their own information, we apply the Kullback–Leibler (KL) divergence as a regularizer $R$ (Sun, Yao et al., 2019). In summary, the general form in Eq. (2) can be modified for our training framework as follows:

$$\begin{aligned}J(\Theta)=&\frac{1}{N_{bs}}\sum_{i=1}^{N_{bs}}\sum_{j=1}^{N_{cls}}\ell\left(C_j^{Base}\left(f\left(\tilde{x}_i;\Theta^{1:l-1}\cup\Theta_j^{l:B}\right)\right);\tilde{y}_i\right)\\&+\frac{1}{2N_{cls}}\sum_{i=1}^{N_{cls}}\sum_{\substack{j=1,\\ j\neq i}}^{N_{cls}}D_{KL}\left(C_i^{Base}\parallel C_j^{Base}\right).\end{aligned} \tag{5}$$

Here, $N_{cls}$ is the number of auxiliary classifiers, and for mathematical simplicity we regard the main classifier as one of the auxiliary classifiers. We use the cross-entropy loss for $\ell$. $\Theta^{1:l-1}\cup\Theta_j^{l:B}$ means that the parameters before the $l$th block are shared among the auxiliary classifiers and the $l:B$ blocks are learned independently for the $j$th classifier.

#### 3.2.3. Discussion

Here, we further discuss the effectiveness of the proposed self-mix and the motivation of auxiliary classifiers as follows.

As regional dropout chooses a random region of an input image and replaces the pixels for other values, it perturbs the data statistics. This prevents the network from memorizing the data statistics of base classes and improves the generalization performance for unseen classes. Meanwhile, there exist two present

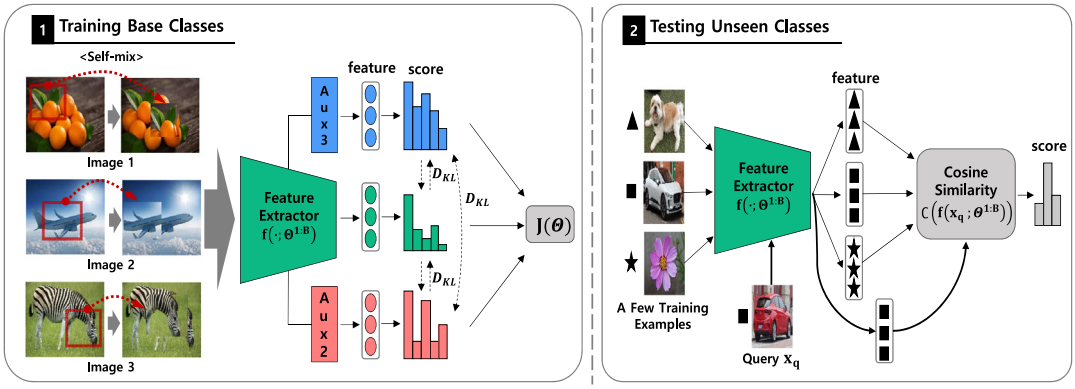


**Fig. 1.** Overview of the proposed self-augmentation framework. The main network consists of three classifiers, two of which are derived from intermediate layers of the main branch. In the training phase, we first apply regional dropout to input images, which removes a part of the image by replacing it with other values. All the classifiers try to learn a more generalizable parameter space by minimizing the cross entropy loss and regularizing their prediction scores to have a similar distribution via the KL divergence. For inference, we simply use the main classifier to evaluate images from unseen classes. The right figure shows the case of 3-way 1-shot learning as an example.

works (DeVries & Taylor, 2017; Yun et al., 2019) as regional dropout techniques and have its own disadvantages. Cutmix (Yun et al., 2019) exchanges two randomly selected patches from two images, thus encouraging the network to learn two labels simultaneously. However, it has been reported that such label smoothing impairs the ability of knowledge distillation (Müller et al., 2019). Considering that our proposed framework employs knowledge distillation, it is less effective for cutmix to exploit the full capacity of our framework. On the other hands, cutout (DeVries & Taylor, 2017) converts the pixels of the region into zeros, which inherently leads to information loss. To solve these problems, we propose self-mix, which exchanges the locations of the patches of an input image itself. Given that self-mix does not have any information loss and label smoothing issues, we find that it generates a synergy effect with knowledge distillation.

Next, several works have found that deep networks are prone to over-confident predictions, and this hinders a network from learning generalization (Pereyra et al., 2017; Thulasidasan et al., 2019; Zhang, Xiang et al., 2018). In other words, it is possible that an over-confident network results in a decision boundary that is sharp (Verma et al., 2018) as highly optimized for the statistics of a training dataset. However, unseen classes are not guaranteed to follow the distribution of training examples for base classes, and a sharp boundary is likely to produce unstable predictions for two slightly different examples of an unseen class. Thus, to alleviate the possible sudden jumps, we employ auxiliary classifiers that share their own information. This helps an optimizer to search for wide valleys (Zhang, Xiang et al., 2018). Also, we found that the explicit regularization about the softmax output produces better generalization ability than regional dropout as an input perturbation.

### 3.3. Local representation learner

We have proposed how to train a network on base classes to produce global representations, which can be generalizable to unseen classes. In the test stage, we have $n$-way $k$-shot training examples and $T$ queries for unseen classes. Thus, we now present how to fine-tune the global representations to yield local representations adjusted for the $n \cdot k$ examples. The overall concept is illustrated in Fig. 2.

#### 3.3.1. Preliminary

For fine-tuning, random transformations are applied on training examples to produce novel weights and fake queries as follows:

$$\{x_1, x_2, \ldots, x_{n\cdot k}\} \xrightarrow[\text{for Training}]{\text{Random Transf.}} \{\tilde{x}_1, \tilde{x}_2, \ldots, \tilde{x}_{n\cdot k}\}$$

$$\{x_1, x_2, \ldots, x_{n\cdot k}\} \xrightarrow[\text{for Fake Queries}]{\text{Random Transf.}} \{\tilde{x}_1^q, \tilde{x}_2^q, \ldots, \tilde{x}_{n\cdot k}^q\},$$

where $\tilde{x}_i$ is used to create a novel weight and $\tilde{x}_i^q$ is used to induce a loss. It is worth noting that we only have access to the $n \cdot k$ examples, and we are never informed about the real queries.

#### 3.3.2. Training

Our objective is not to destroy the well-learned global representations and we promise to be more discriminative after fine-tuning. Thus, we clone the last block of the pre-trained network and only fine-tune the cloned block. The features extracted from the separate networks are denoted as

$$f_i^{Global} := f\left(\tilde{x}_i; \Theta_{pre}^{1:B}\right)$$
$$f_i^{Bias} := f\left(\tilde{x}_i; \Theta_{pre}^{1:B-1} \cup \Theta^B\right),$$

where $\Theta_{pre}$ denotes the pre-trained parameters for the base classes. Local representation is defined as the sum of the above two features. Similarly, the features for queries can be defined as $f_{i,q}^{Global}$ and $f_{i,q}^{Bias}$. Then, according to Eq. (3), the weight for the $j$th unseen class is produced by

$$w_j^N = \frac{1}{k}\sum_{i=1}^{k}\left(f_{i,j}^{Global} + f_{i,j}^{Bias}\right). \tag{6}$$

As we have formed novel weights and features for fake queries, a cost function can be defined as

$$J(\Theta) = \frac{1}{n\cdot k}\sum_{i=1}^{n\cdot k}\ell\left(C^{Novel}\left(f_{i,q}^{Global} + f_{i,q}^{Bias}\right); \tilde{y}_i\right) + \gamma\sum_{j=1}^{n}\sum_{i=1}^{k}\|f_{i,j}^{Global} - f_{i,j}^{Bias}\|_2, \tag{7}$$

where the regularizer $\gamma$ prevents the fine-tuned block $\Theta^B$ from destroying the well-learned feature space given that only a few training examples are available. Overall, we try to learn the bias term to increase the distance between classes that are close to each other so that they are more distinguishable. We conceptualize the effect of the bias representation in Fig. 3.

#### 3.3.3. Inference

A query is classified by the trivial softmax output, but this time we use $T$ real queries. Our proposed fine-tuning method can be applied to any global representations trained on base classes.

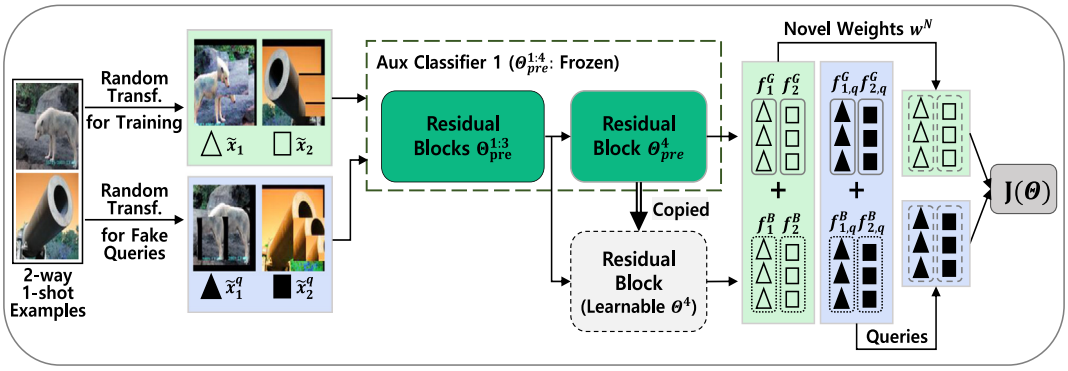


**Fig. 2.** Overview of our local representation learning procedure. We conceptualize our method with an example of 2-way 1-shot learning. To fine-tune a network, we randomly apply transformations such as random crop, horizontal flip and regional dropout twice. The first application is intended to produce novel weights $w^N$, and the other is to generate fake queries. These two features yield a loss for a network to be fine-tuned. We copy the last convolutional block of the network and the parameters of the block will be learned to generate a bias $f_i^B$. The role of this bias term is to separate classes that are close to each other.

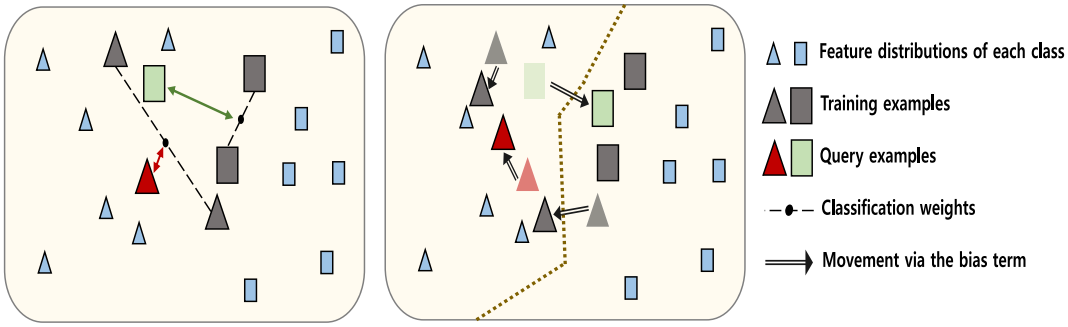


**Fig. 3.** Effect of the bias representation. (Left) The feature extractor trained on base classes cannot locate unseen classes correctly in some cases such as the squares. (Right) The bias representation extracted from a residual block fine-tuned on unseen classes can adjust the features to be more distinguishable.

### 3.4. Experimental setup

#### 3.4.1. Datasets

*Mini*ImageNet (Vinyals et al., 2016) consists of 100 classes randomly selected from ILSVRC-2012 (Russakovsky et al., 2015) and each class has 600 images, each sized 84 × 84. We follow the split proposed in Ravi and Larochelle (2017), namely 64, 16 and 20 classes for training, validation and testing, respectively. *Tiered*ImageNet (Ren et al., 2018) has 608 classes randomly selected from ILSVRC-2012 (Russakovsky et al., 2015) and these classes are grouped into 34 higher level categories. They are then split into 20, 6 and 8 categories to further build 341, 91 and 160 classes for training, validation and testing, respectively. A much larger number of images (totally 779165 images) are sized 84 × 84. CIFAR-FS (Bertinetto, Henriques, Torr, & Vedaldi, 2019) consists of 100 classes randomly selected from CIFAR-100 (Krizhevsky, Nair, & Hinton, 2009) and each class has 600 images, each of size 32 × 32. The classes are split into 64, 16 and 20 classes for training, validation and testing, respectively. CUB-200-2011 (Wah, Branson, Welinder, Perona, & Belongie, 2011) consists of 200 classes and we follow the split proposed in Ye et al. (2020) that all classes are divided into 100, 50 and 50 classes for training, validation and testing, respectively.

#### 3.4.2. Evaluation

We report the performance averaged over 2000 randomly sampled tasks from the test set to obtain the 95% confidence interval. We use $T = 15$ test queries for the 5-way 5-shot and the 5-way 1-shot, as in Ravi and Larochelle (2017), Snell et al. (2017) and Vinyals et al. (2016).

#### 3.4.3. Implementation details

For all the datasets, we report the results using ResNet-12 (Lee, Maji, Ravichandran, & Soatto, 2019), which has four blocks. Each block consists of three 3 × 3 Convolution-BatchNorm-LeakyReLU (0.1) and one 2 × 2 max pooling. The depths of the four blocks are 64 → 160 → 320 → 640. We trained a network for 60 epochs (each epoch consisted of 1000 iterations). Initial learning rate was 0.1 and decreased to 0.006, 0.0012 and 0.00024 at 20, 40 and 50 epochs, respectively. Only in *tiered*ImageNet, the network was trained for 100 epochs (each epoch consisted of 2000 iterations). Initial learning rate was 0.1 and decreased to 0.006, 0.0012 and 0.00024 at 40, 80 and 90 epochs, respectively.

For self-mix, we randomly sampled a cropping region $P_1 = (r_{x_1}, r_{y_1}, r_w, r_h)$ from an image. Length of the patch $(r_w, r_h)$ was set to $(\frac{W}{2}, \frac{H}{2})$. Then, a patch $P_2$ from the same input was sampled with randomly chosen $(r_{x_2}, r_{y_2}) \left(\neq (r_{x_1}, r_{y_1})\right)$ and the same $(r_w, r_h)$. The code-level description is shown in Algorithm 1. For self-distillation, auxiliary classifiers are branched from the 2nd and 3rd blocks of ResNet-12. The two auxiliary classifiers have two and one new ResNet blocks, respectively. All branches were initialized independently, which forces the network into learning different posterior distributions. We use stochastic gradient descent (SGD) with a Nesterov momentum of 0.9 and a weight decay of 0.0005. We fix the scale parameter for the classifier to $\tau = 20$. For the local representation learner (LRL), we used the SGD optimizer and the model was trained for 200 epochs per task. The initial learning rates used for our experiments are shown in Table 1 and were decreased by a factor of 10 at 80, 120, 160 epochs. To generate fake queries and novel weights, we applied horizontal flip, random crop, color jittering and then we further applied regional dropout such as self-mix.

**Table 1**
Initial learning rates and regularization parameters for the local representation learner.

| Method | *mini*ImageNet | | *tiered*ImageNet | | CIFAR-FS | | CUB | | *mini*ImageNet → CUB | |
|---|---|---|---|---|---|---|---|---|---|---|
| | 1-shot | 5-shot | 1-shot | 5-shot | 1-shot | 5-shot | 1-shot | 5-shot | 1-shot | 5-shot |
| Learning rate | 1e−2 | 1e−1 | 1e−2 | 1e−1 | 1e−3 | 1e−2 | 1e−4 | 1e−2 | 1e−2 | 1e−1 |
| Regularizer | 1e−1 | 1e−1 | 1e−1 | 1e−1 | 1e−2 | 1e−2 | 1e−3 | 1e−2 | 1e−1 | 1e−1 |

**Table 2**
5-way few-shot classification accuracies on *mini*ImageNet and *tiered*ImageNet with 95% confidence intervals. All accuracy results are averaged over 2000 tasks randomly sampled from the test set. LRL denotes the local representation learner. '-' indicates that the performances were not reported by the papers.

| Method | Backbone | *mini*ImageNet | | *tiered*ImageNet | |
|---|---|---|---|---|---|
| | | 1-shot | 5-shot | 1-shot | 5-shot |
| MAML (Finn et al., 2017) | C64F | 48.70 ± 1.84 | 63.11 ± 0.92 | – | - |
| ProtoNet (Snell et al., 2017) | C64F | 49.42 ± 0.54 | 68.20 ± 0.66 | – | - |
| RelationNet (Sung et al., 2018) | C64F | 50.44 ± 0.82 | 65.32 ± 0.66 | – | - |
| R2D2 (Bertinetto et al., 2019) | C512F | 51.80 ± 0.20 | 68.40 ± 0.20 | – | - |
| LEO (Rusu et al., 2019) | WRN-28-10 | 61.76 ± 0.08 | 77.59 ± 0.12 | 66.33 ± 0.05 | 81.44 ± 0.09 |
| MTL (Sun, Liu et al., 2019) | ResNet12 | 61.20 ± 1.80 | 75.50 ± 0.80 | – | – |
| AM3-TADAM (Xing, Rostamzadeh, Oreshkin, & Pinheiro, 2019) | ResNet12 | 65.30 ± 0.49 | 78.10 ± 0.36 | 69.08 ± 0.47 | 82.58 ± 0.31 |
| MetaOptNet (Lee et al., 2019) | ResNet12 | 62.64 ± 0.61 | 78.63 ± 0.46 | 65.99 ± 0.72 | 81.56 ± 0.53 |
| DC (Lifchitz et al., 2019) | ResNet12 | 62.53 ± 0.19 | 78.95 ± 0.19 | – | – |
| CAM (Hou, Chang, Bingpeng, Shan, & Chen, 2019) | ResNet12 | 63.85 ± 0.48 | 79.44 ± 0.33 | 69.89 ± 0.51 | 84.23 ± 0.37 |
| CC+Rotation (Gidaris et al., 2019) | WRN-28-10 | 62.93 ± 0.45 | 79.87 ± 0.33 | 70.53 ± 0.51 | 84.98 ± 0.36 |
| CTM (Li et al., 2019) | ResNet18 | 64.12 ± 0.82 | 80.51 ± 0.13 | 68.41 ± 0.39 | 84.28 ± 1.73 |
| Robust 20-dist++ (Dvornik et al., 2019) | ResNet18 | 63.73 ± 0.62 | 81.19 ± 0.43 | 70.44 ± 0.32 | 85.43 ± 0.21 |
| DeepEMD (Zhang et al., 2020) | ResNet12 | 65.91 ± 0.82 | 82.41 ± 0.56 | 71.16 ± 0.87 | 86.03 ± 0.58 |
| FEAT (Ye et al., 2020) | ResNet12 | **66.78 ± 0.20** | 82.05 ± 0.14 | 70.80 ± 0.23 | 84.79 ± 0.16 |
| Self-augmentation | ResNet12 | 65.27 ± 0.45 | 81.84 ± 0.32 | 71.26 ± 0.50 | 85.55 ± 0.34 |
| Self-augmentation + LRL | ResNet12 | 65.37 ± 0.45 | **82.68 ± 0.30** | **71.31 ± 0.50** | **86.41 ± 0.30** |

**Algorithm 1** Pseudo-Code of Self-Mix

**Input** Image with size C× W× H
**Length** Patch size

**function** SELFMIX(Input, Length)
  H = Input.size(2)
  W = Input.size(1)
  $x$ = randint(0,W)
  $y$ = randint(0,H)
  $x_1$ = Clip($x$ - Length/2, 0, W)
  $x_2$ = Clip($x$ + Length/2, 0, W)
  $y_1$ = Clip($y$ - Length/2, 0, H)
  $y_2$ = Clip($y$ + Length/2, 0, H)
  **while** true **do**
    $x_n$ = randint(0+$(x_2 - x_1)$/2,W-$(x_2 - x_1)$/2)
    $y_n$ = randint(0+$(y_2 - y_1)$/2,H-$(y_2 - y_1)$/2)
    **if** $y_n! = y_1$ or $x_n! = x_1$ **then**
      break;
    **end if**
  **end while**
  Input[:,$x_1 : x_2, y_1 : y_2$] = Input[:, $x_n : x_n + (x_2 - x_1), y_n : y_n + (y_2 - y_1)$]
**end function**

### 3.5. Comparison with the state-of-the-art methods

We compare the proposed method with the state-of-the-art algorithms. As shown in Table 2, recent techniques (Li et al., 2019; Xing et al., 2019) perform well in certain environments such as 1-shot or 5-shot, or on a certain dataset, while the proposed method works decently in all settings. However, as DeepEMD (Zhang et al., 2020) and FEAT (Ye et al., 2020) produce a better performance for one-shot learning on *mini*ImageNet, we analyze the reason as follows.

DeepEMD (Zhang et al., 2020) extracts the feature maps from quary and training images and measures element-wise similarities ($HW \times HW$) using the so-called Earth Mover's Distance. FEAT (Ye et al., 2020) aims to exploit the mutual relationships among training images by applying the well-known Transformer architecture (Vaswani et al., 2017), which results in an input-adaptive feature extractor as the output of the Transformer is affected by the combination of the training images given for a current task. In other words, they inherently employ an attention mechanism to cope with inter-class variation as DeepEMD (Zhang et al., 2020) exploits local features (e.g., element-wise comparisons between the feature maps of query and training examples) and FEAT (Ye et al., 2020) uses a memory-like architecture (e.g., Transformer Vaswani et al., 2017). Thus, in the case of one-shot learning, the attentive feature is advantageous to remove background or noisy information.

Nonetheless, our method is superior to DeepEMD (Zhang et al., 2020) and FEAT (Ye et al., 2020) in all other settings, and we stress that they lack the ability to handle fine-grained information. To be specific, as shown in Table 3, self-augmentation significantly outperforms all the state-of-the-art methods by a large margin for the CUB dataset. This indicates that the input–output perturbation approach during training base classes is more effective to distinguish inter-class variation on unseen classes than capturing local information (Zhang et al., 2020) or the commonality among training examples (Ye et al., 2020).

To further validate the effectiveness of our idea, we lastly evaluate the few-shot performance on CIFAR-FS in Table 4. To summarize, our method consistently outperforms the state-of-the arts on various datasets. This indicates that it is worthwhile investigating the generalization ability of the standard supervision in relation to few-shot learning.

### 3.6. Domain shift: miniImageNet to CUB

We further analyze the generalization ability and the network calibration of the proposed framework. After training a network on *mini*ImageNet, we perform 5-way classification on CUB (Wah et al., 2011). This is a challenging problem as (1) CUB is designed for fine-grained image classification with 200 bird species, (2) the distributions of the two datasets are largely different and (3) we

**Table 3**
5-way few-shot classification accuracies on CUB with the 95% confidence intervals.

| Method | CUB | |
|---|---|---|
| | 1-shot | 5-shot |
| RelationNet[a] (Sung et al., 2018) | 66.20 ± 0.99 | 82.30 ± 0.58 |
| ProtoNet[a] (Snell et al., 2017) | 66.09 ± 0.92 | 82.50 ± 0.58 |
| Cosine classifier[a] (Chen et al., 2019) | 67.30 ± 0.86 | 84.75 ± 0.60 |
| DeepEMD (Zhang et al., 2020) | 75.65 ± 0.83 | 88.69 ± 0.50 |
| FEAT (Ye et al., 2020) | 68.87 ± 0.22 | 82.90 ± 0.15 |
| Self-augmentation | 78.11 ± 0.42 | **92.47 ± 0.20** |
| Self-augmentation + LRL | **78.21 ± 0.44** | 92.39 ± 0.21 |

[a]Results from Zhang et al. (2020).

**Table 4**
5-way few-shot classification accuracies on CIFAR-FS with the 95% confidence intervals.

| Method | CIFAR-FS | |
|---|---|---|
| | 1-shot | 5-shot |
| MAML[a] (Finn et al., 2017) | 58.90 ± 1.90 | 71.50 ± 1.00 |
| ProtoNet[a] (Snell et al., 2017) | 55.50 ± 0.70 | 72.00 ± 0.60 |
| RelationNet[a] (Sung et al., 2018) | 55.00 ± 1.00 | 69.30 ± 0.80 |
| R2D2 (Bertinetto et al., 2019) | 65.30 ± 0.20 | 79.40 ± 0.10 |
| MetaOptNet (Lee et al., 2019) | 72.00 ± 0.70 | 84.20 ± 0.50 |
| CC+Rotation (Gidaris et al., 2019) | 73.62 ± 0.31 | 86.05 ± 0.22 |
| Self-augmentation | 76.91 ± 0.46 | 89.31 ± 0.31 |
| Self-augmentation + LRL | **76.95 ± 0.46** | **89.54 ± 0.39** |

[a]Results from Bertinetto et al. (2019).

**Table 5**
5-way few-shot classification accuracies on the domain shift (*mini*ImageNet → CUB) with the 95% confidence intervals. '-' denotes that the performance is not provided by the study.

| Method | *mini*ImageNet → CUB | |
|---|---|---|
| | 1-shot | 5-shot |
| RelationNet[a] (Sung et al., 2018) | 36.86 ± 0.70 | 57.71 ± 0.73 |
| ProtoNet[a] (Snell et al., 2017) | 41.36 ± 0.70 | 62.02 ± 0.70 |
| Linear classifier[a] (Chen et al., 2019) | 44.33 ± 0.74 | 65.57 ± 0.70 |
| Cosine classifier[a] (Chen et al., 2019) | 44.51 ± 0.80 | 62.04 ± 0.76 |
| Diverse 20 Full (Dvornik et al., 2019) | – | 66.17 ± 0.55 |
| Self-augmentation | 51.50 ± 0.46 | 72.00 ± 0.39 |
| Self-augmentation + LRL | **51.65 ± 0.46** | **74.20 ± 0.37** |

[a]We re-implemented the official code (Chen et al., 2019) to evaluate 1-shot accuracies, and 5-shot accuracies were reported from Chen et al. (2019).

only have 1 or 5 training examples for few-shot learning. With those difficulties, Table 5 shows that self-augmentation significantly surpasses the previous works (Chen et al., 2019; Dvornik et al., 2019; Snell et al., 2017; Sung et al., 2018).

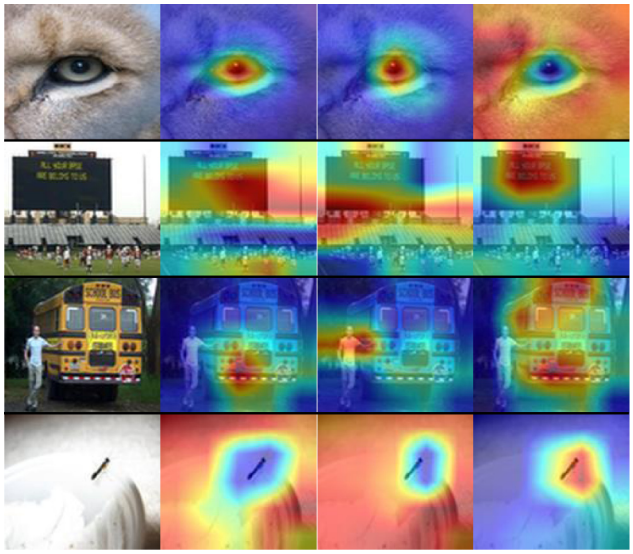


**Fig. 4.** Visualization using the class activation map (Zhou, Khosla, Lapedriza, Oliva, & Torralba, 2016) to show the regions that deep networks focus on.

### 3.7. Ablation study

#### 3.7.1. Effect of the local representation learner

Fig. 4 shows that there exist cases where the local representation learner (LRL) fixes the deep network to focus on more discriminative parts. As a result, only self-augmentation with LRL correctly classifies the images. This indicates that a network can be further enhanced even with a few training examples using a carefully designed strategy.

#### 3.7.2. Comparison with various regional dropout techniques

As we proposed the framework consolidating regional dropout and self-distillation, Table 6 shows that how performance changes by adopting various regional dropout methods and self-distillation. Baseline refers to a network using light augmentation such as random color jittering, cropping and horizontal flipping. The results indicate four notable aspects: (1) Self-augmentation significantly outperforms the baseline using light augmentation only. (2) Although either regional dropout or self-distillation can improve the generalization capability, exploiting both methods leads to higher performance gains. (3) As discussed in Section 3.2.1, the proposed self-mix has a synergistic effect with

**Table 6**
Ablation study on *mini*ImageNet, *tiered*ImageNet and cross-domain benchmarks. Baseline refers to a vanilla network without any regional dropout techniques. SD and SA denote self-distillation and self-augmentation, respectively.

| Method | *mini*ImageNet | | *tiered*ImageNet | | *mini*ImageNet → CUB | |
|---|---|---|---|---|---|---|
| | 1-shot | 5-shot | 1-shot | 5-shot | 1-shot | 5-shot |
| Baseline | 61.42 ± 0.45 | 78.32 ± 0.33 | 68.22 ± 0.50 | 83.21 ± 0.36 | 47.76 ± 0.44 | 67.40 ± 0.38 |
| Cutout | 62.38 ± 0.44 | 79.18 ± 0.33 | 69.40 ± 0.51 | 84.27 ± 0.36 | 47.46 ± 0.44 | 67.79 ± 0.40 |
| Cutmix | 62.81 ± 0.45 | 79.82 ± 0.33 | 69.09 ± 0.49 | 84.21 ± 0.35 | 48.35 ± 0.44 | 67.77 ± 0.39 |
| Selfmix | 62.85 ± 0.45 | 79.83 ± 0.32 | 69.95 ± 0.40 | 84.39 ± 0.35 | 48.73 ± 0.45 | 69.20 ± 0.39 |
| Self-Distillation | 63.11 ± 0.45 | 79.93 ± 0.33 | 70.05 ± 0.49 | 84.92 ± 0.34 | 48.91 ± 0.44 | 69.45 ± 0.38 |
| SD + Cutout | 64.61 ± 0.44 | 81.57 ± 0.31 | 70.76 ± 0.50 | 85.50 ± 0.35 | 48.94 ± 0.43 | 69.65 ± 0.39 |
| SD + Cutout + LRL | 64.93 ± 0.45 | 82.34 ± 0.30 | 70.82 ± 0.50 | 86.15 ± 0.33 | 48.93 ± 0.42 | 73.37 ± 0.36 |
| SD + Cutmix | 64.44 ± 0.45 | 81.58 ± 0.32 | 70.46 ± 0.49 | 85.51 ± 0.34 | 50.43 ± 0.45 | 70.70 ± 0.39 |
| SD + Cutmix + LRL | 64.67 ± 0.45 | 81.52 ± 0.31 | 70.48 ± 0.48 | 85.60 ± 0.34 | 49.88 ± 0.43 | 72.35 ± 0.37 |
| **Self-augmentation** | **65.27 ± 0.45** | **81.84 ± 0.32** | **71.26 ± 0.50** | **85.55 ± 0.34** | **51.50 ± 0.46** | **72.00 ± 0.39** |
| **SA +LRL** | **65.37 ± 0.45** | **82.68 ± 0.30** | **71.31 ± 0.50** | **86.41 ± 0.33** | **51.65 ± 0.46** | **74.20 ± 0.37** |

**Table 7**
Effect of label smoothing on *mini*ImageNet. Applying label smoothing to each method decreases their original performances for unseen classes.

| Method | Baseline | Baseline | | Self-distillation | | Self-augmentation | |
|---|---|---|---|---|---|---|---|
| | | 1-shot | 5-shot | 1-shot | 5-shot | 1-shot | 5-shot |
| Test class | Base class | Unseen class | | | | | |
| w/o label smoothing | 80.22 | 61.42 | 78.32 | 63.11 | 79.93 | 65.27 | 81.84 |
| w label smoothing | 81.36 | 61.27 | 77.03 | 61.96 | 77.45 | 63.29 | 78.24 |
| Gain | (+1.14) | (−0.15) | (−1.29) | (−1.15) | (−2.48) | (−1.84) | (−3.60) |

**Table 8**
5-way few-shot classification accuracies with various data augmentation methods on *mini*ImageNet.

| Method | *mini*ImageNet | |
|---|---|---|
| | 1-shot | 5-shot |
| Horizontal flipping | 58.91 ± 0.44 | 76.04 ± 0.33 |
| Color jittering | 55.58 ± 0.44 | 72.59 ± 0.34 |
| Random cropping | 58.82 ± 0.44 | 75.93 ± 0.33 |
| Baseline[a] | 61.42 ± 0.45 | 78.32 ± 0.33 |
| MixUp (Zhang, Cisse et al., 2018) | 61.26 ± 0.45 | 78.89 ± 0.32 |
| RandAugment (Cubuk, Zoph, Shlens, & Le, 2020) | 61.45 ± 0.45 | 78.50 ± 0.35 |
| Self-Mix | 62.85 ± 0.45 | 79.83 ± 0.32 |

[a]Baseline refers to the network using horizontal flipping, color jittering and random cropping.

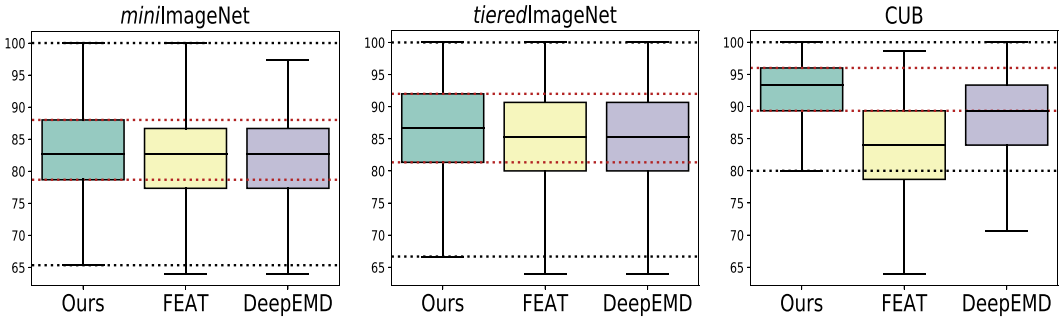


**Fig. 5.** Statistical analyses using box plots to compare the three competing methods.

self-distillation as it does not require pixel removal (DeVries & Taylor, 2017) or mixed labels (Yun et al., 2019). (4) When using cutmix (Yun et al., 2019) for the local representation learner, the performance remains almost the same. As only a few training examples exist, we conjecture that the mixed labels produced by cutmix increase the complexity of fine-tuning. To sum up, although several regional dropout techniques have been studied, self-mix is more flexible to be used with distillation or local representation leaning.

### 3.7.3. Effect of label smoothing

As we deal with a memorization problem of deep networks in terms of few-shot learning, we further present the performance with label smoothing that is another way to perturb output distributions. Though it is well-known that label smoothing is beneficial for standard classification problems (Szegedy et al., 2015), Table 7 indicates that label smoothing is not effective for few-shot learning and there exist a significant performance drop with self-distillation. Furthermore, it is worth noting that Table 6 also shows that cutmix, which learns two labels simultaneously similar to label smoothing, has less performance gain when using self-distillation.

### 3.7.4. Comparison with various data augmentation

In Table 8, we further provide the comparison with various input-level data augmentation methods such as RandAugment (Cubuk et al., 2020) that is an automated augmentation strategy and MixUp (Zhang, Cisse et al., 2018) that performs a convex combination between two images and their labels. Although it has been known that RandAugment (Cubuk et al., 2020) and MixUp (Zhang, Cisse et al., 2018) improve the standard classification performances, Table 8 shows that the approaches do not improve the few-shot performances compared to the proposed self-mix. we conjecture that as deep networks aggregate local information (e.g., using the global pooling), it is more crucial to agitate the local information such as the parts of an object shape than to manipulate the entire image while maintaining the shape of the object.

### 3.7.5. Statistical analysis

As the proposed method achieves the competitive performances to DeepEMD (Zhang et al., 2020) and FEAT (Ye et al., 2020) on *mini*ImageNet and *tiered*ImageNet, we further provide statistical comparisons using the box plot. As shown in Fig. 5, the proposed method has comparable medians but produces higher accuracies at the maximum, Q1, Q3 and the minimum. Furthermore, for CUB, we can clearly observe that the gap between the maximum and the minimum is considerably small, which demonstrates the statistical stability compared to the others.

### 3.7.6. Number of classifiers

Fig. 6 shows that how different numbers of classifiers $N_{cls}$ in Eq. (5) affect the classification performance. We can verify that there exists an optimal number of classifiers and this can be seen as the trade-off between the amount of knowledge sharing and the complexity of the parameter space.

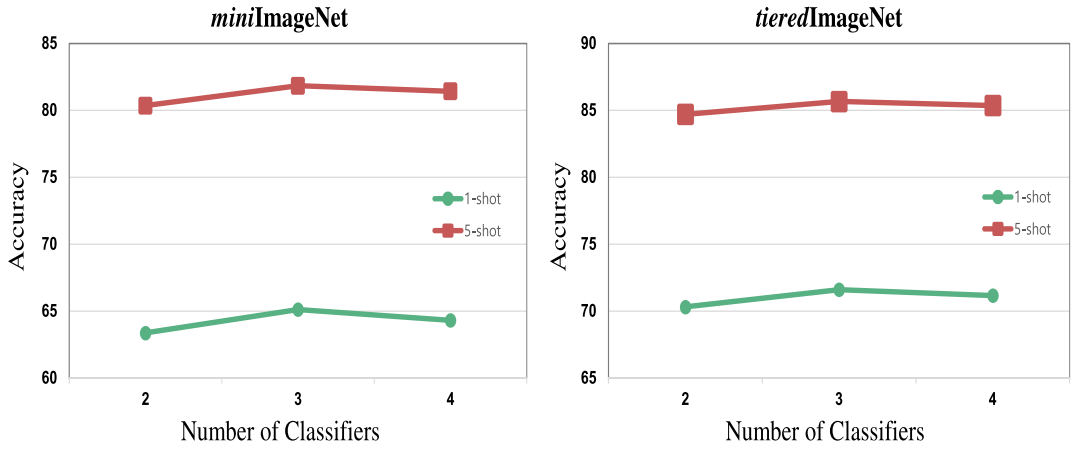


**Fig. 6.** Test accuracies (%) with various numbers of classifiers for self-distillation. In both cases, using three classifiers shows the highest accuracy.

## 4. Conclusion

In this paper, we show that unseen classes with a few training examples can be classified with a standard supervised training. Especially, we aim at generalizing deep networks to unseen classes by alleviating the memorization phenomenon, which is less studied for few-shot learning. To achieve this, we design a framework using regional dropout and self-distillation to perturb the input and output information. Especially, we show that the newly proposed regional dropout, called self-mix, produces state-of-the-art results when cooperating with self-distillation. We also present a local representation learner to exploit a few training examples of unseen classes, which improves the performance for all few-shot learning benchmarks and especially works well on a cross-domain task. More importantly, we show that existing perturbation methods, which are designed for a standard classification setting, such as cutmix, cutout and label smoothing are not the optimal choices for few-shot learning as they are not flexible enough to be used with knowledge distillation or local representation leaning.

## Declaration of competing interest

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

## Acknowledgments

This work was supported by Institute for Information & communications Technology Planning & Evaluation (IITP) grant funded by the Korea government (MSIT) (No. 2017-0-01779, A machine learning and statistical inference framework for explainable artificial intelligence, No. 2019-0-01371, Development of brain-inspired AI with human-like intelligence, and No. 2019-0-00079, Artificial Intelligence Graduate School Program, Korea University).

## References

Bertinetto, L., Henriques, J. F., Torr, P., & Vedaldi, A. (2019). Meta-learning with differentiable closed-form solvers. In *International conference on learning representations* (pp. 1–15).

Bulthoff, H. H., Lee, S.-W., Poggio, T., & Wallraven, C. (2003). *Biologically motivated computer vision*. Springer-Verlag.

Chen, W.-Y., Liu, Y.-C., Kira, Z., Wang, Y.-C. F., & Huang, J.-B. (2019). A closer look at few-shot classification. In *International conference on learning representations* (pp. 1–16).

Cubuk, E. D., Zoph, B., Shlens, J., & Le, Q. V. (2020). Randaugment: Practical automated data augmentation with a reduced search space. In *Proceedings of the IEEE conference on computer vision and pattern recognition workshops* (pp. 702–703).

DeVries, T., & Taylor, G. W. (2017). Improved regularization of convolutional neural networks with cutout. arXiv preprint arXiv:1708.04552.

Dvornik, N., Schmid, C., & Mairal, J. (2019). Diversity with cooperation: Ensemble methods for few-shot classification. In *Proceedings of the IEEE international conference on computer vision* (pp. 3723–3731).

Finn, C., Abbeel, P., & Levine, S. (2017). Model-agnostic meta-learning for fast adaptation of deep networks. In *Proceedings of international conference on machine learning* (pp. 1126–1135).

Gidaris, S., Bursuc, A., Komodakis, N., Pérez, P., & Cord, M. (2019). Boosting few-shot visual learning with self-supervision. In *Proceedings of the IEEE international conference on computer vision* (pp. 8059–8068).

Gidaris, S., & Komodakis, N. (2018). Dynamic few-shot visual learning without forgetting. In *Proceedings of the IEEE conference on computer vision and pattern recognition* (pp. 4367–4375).

Gidaris, S., & Komodakis, N. (2019). Generating classification weights with gnn denoising autoencoders for few-shot learning. In *Proceedings of the IEEE conference on computer vision and pattern recognition* (pp. 21–30).

Guo, C., Pleiss, G., Sun, Y., & Weinberger, K. Q. (2017). On calibration of modern neural networks. In *Proceedings of international conference on machine learning* (pp. 1321–1330).

He, K., Zhang, X., Ren, S., & Sun, J. (2016). Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition* (pp. 770–778).

Hinton, G., Vinyals, O., & Dean, J. (2015). Distilling the knowledge in a neural network. arXiv preprint arXiv:1503.02531.

Hou, R., Chang, H., Bingpeng, M., Shan, S., & Chen, X. (2019). Cross attention network for few-shot classification. In *Advances in neural information processing systems* (pp. 4005–4016).

Kang, D., Han, H., Jain, A. K., & Lee, S.-W. (2014). Nighttime face recognition at large standoff: Cross-distance and cross-spectral matching. *Pattern Recognition*, *47*(12), 3750–3766.

Krizhevsky, A., Nair, V., & Hinton, G. (2009). Cifar-10 and cifar-100 datasets. URl: https://www.cs.toronto.edu/kriz/cifar.html 6.

Lan, X., Zhu, X., & Gong, S. (2018). Knowledge distillation by on-the-fly native ensemble. In *Advances in neural information processing systems* (pp. 7517–7527).

Lee, S.-W., & Kim, S.-Y. (1999). Integrated segmentation and recognition of handwritten numerals with cascade neural network. *IEEE Transactions on Systems, Man, and Cybernetics*, *29*(2), 285–290.

Lee, K., Maji, S., Ravichandran, A., & Soatto, S. (2019). Meta-learning with differentiable convex optimization. In *Proceedings of the IEEE conference on computer vision and pattern recognition* (pp. 10657–10665).

Li, H., Eigen, D., Dodge, S., Zeiler, M., & Wang, X. (2019). Finding task-relevant features for few-shot learning by category traversal. In *Proceedings of the IEEE conference on computer vision and pattern recognition* (pp. 1–10).

Lifchitz, Y., Avrithis, Y., Picard, S., & Bursuc, A. (2019). Dense classification and implanting for few-shot learning. In *Proceedings of the IEEE conference on computer vision and pattern recognition* (pp. 9258–9267).

Liu, Y., Gao, X., Gao, Q., Han, J., & Shao, L. (2020). Label-activating framework for zero-shot learning. *Neural Networks*, *121*, 1–9.

Müller, R., Kornblith, S., & Hinton, G. (2019). When does label smoothing help?. In *Advances in neural information processing systems* (pp. 4694–4703).

Neyshabur, B., Bhojanapalli, S., McAllester, D., & Srebro, N. (2017). Exploring generalization in deep learning. In *Advances in neural information processing systems* (pp. 5947–5956).
Oreshkin, B., López, P. R., & Lacoste, A. (2018). Tadam: Task dependent adaptive metric for improved few-shot learning. In *Advances in neural information processing systems* (pp. 721–731).
Pereyra, G., Tucker, G., Chorowski, J., Kaiser, Ł., & Hinton, G. (2017). Regularizing neural networks by penalizing confident output distributions. arXiv preprint arXiv:1701.06548.
Qi, H., Brown, M., & Lowe, D. G. (2018). Low-shot learning with imprinted weights. In *Proceedings of the IEEE conference on computer vision and pattern recognition* (pp. 5822–5830).
Ravi, S., & Larochelle, H. (2017). Optimization as a model for few-shot learning. In *International conference on learning representations* (pp. 1–11).
Ren, M., Triantafillou, E., Ravi, S., Snell, J., Swersky, K., Tenenbaum, J. B., et al. (2018). Meta-learning for semi-supervised few-shot classification. In *International conference on learning representations* (pp. 1–15).
Roh, M.-C., Kim, T.-Y., Park, J., & Lee, S.-W. (2007). Accurate object contour tracking based on boundary edge selection. *Pattern Recognition*, *40*(3), 931–943.
Roh, M.-C., Shin, H.-K., & Lee, S.-W. (2010). View-independent human action recognition with volume motion template on single stereo camera. *Pattern Recognition Letters*, *31*(7), 639–647.
Russakovsky, O., Deng, J., Su, H., Krause, J., Satheesh, S., Ma, S., et al. (2015). Imagenet large scale visual recognition challenge. *International Journal of Computer Vision*, *115*(3), 211–252.
Rusu, A. A., Rao, D., Sygnowski, J., Vinyals, O., Pascanu, R., Osindero, S., et al. (2019). Meta-learning with latent embedding optimization. In *International conference on learning representations* (pp. 1–17).
Snell, J., Swersky, K., & Zemel, R. (2017). Prototypical networks for few-shot learning. In *Advances in neural information processing systems* (pp. 4077–4087).
Sun, Q., Liu, Y., Chua, T.-S., & Schiele, B. (2019). Meta-transfer learning for few-shot learning. In *Proceedings of the IEEE conference on computer vision and pattern recognition* (pp. 403–412).
Sun, D., Yao, A., Zhou, A., & Zhao, H. (2019). Deeply-supervised knowledge synergy. In *Proceedings of the IEEE conference on computer vision and pattern recognition* (pp. 6997–7006).
Sung, F., Yang, Y., Zhang, L., Xiang, T., Torr, P. H., & Hospedales, T. M. (2018). Learning to compare: Relation network for few-shot learning. In *Proceedings of the IEEE conference on computer vision and pattern recognition* (pp. 1199–1208).
Szegedy, C., Liu, W., Jia, Y., Sermanet, P., Reed, S., Anguelov, D., et al. (2015). Going deeper with convolutions. In *Proceedings of the IEEE conference on computer vision and pattern recognition* (pp. 1–9).
Thulasidasan, S., Chennupati, G., Bilmes, J., Bhattacharya, T., & Michalak, S. (2019). On mixup training: Improved calibration and predictive uncertainty for deep neural networks. In *Advances in neural information processing systems* (pp. 13888–13899).
Tokozume, Y., Ushiku, Y., & Harada, T. (2018). Between-class learning for image classification. In *Proceedings of the IEEE conference on computer vision and pattern recognition* (pp. 5486–5494).
Ukita, J. (2020). Causal importance of low-level feature selectivity for generalization in image recognition. *Neural Networks*, *125*, 185–193.
Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A. N., et al. (2017). Attention is all you need. In *Advances in neural information processing systems* (pp. 5998–6008).
Verma, V., Lamb, A., Beckham, C., Najafi, A., Mitliagkas, I., Courville, A., et al. (2018). Manifold mixup: Better representations by interpolating hidden states. In *Proceedings of international conference on machine learning* (pp. 6438–6447).
Vinyals, O., Blundell, C., Lillicrap, T., Wierstra, D., et al. (2016). Matching networks for one shot learning. In *Advances in neural information processing systems* (pp. 3630–3638).
Wah, C., Branson, S., Welinder, P., Perona, P., & Belongie, S. (2011). The caltech-ucsd birds-200-2011 dataset. URl: http://www.vision.caltech.edu/visipedia/CUB-200-2011.html.
Xing, C., Rostamzadeh, N., Oreshkin, B., & Pinheiro, P. O. (2019). Adaptive cross-modal few-shot learning. In *Advances in neural information processing systems* (pp. 4848–4858).
Ye, H.-J., Hu, H., Zhan, D.-C., & Sha, F. (2020). Few-shot learning via embedding adaptation with set-to-set functions. In *Proceedings of the IEEE conference on computer vision and pattern recognition* (pp. 8808–8817).
Yun, S., Han, D., Oh, S. J., Chun, S., Choe, J., & Yoo, Y. (2019). Cutmix: Regularization strategy to train strong classifiers with localizable features. In *Proceedings of the IEEE international conference on computer vision* (pp. 6023–6032).
Zhang, C., Bengio, S., Hardt, M., Recht, B., & Vinyals, O. (2017). Understanding deep learning requires rethinking generalization. In *International conference on learning representations* (pp. 1–15).
Zhang, C., Cai, Y., Lin, G., & Shen, C. (2020). DeepEMD: Few-shot image classification with differentiable earth mover's distance and structured classifiers. In *Proceedings of the IEEE conference on computer vision and pattern recognition* (pp. 12203–12213).
Zhang, H., Cisse, M., Dauphin, Y. N., & Lopez-Paz, D. (2018). Mixup: Beyond empirical risk minimization. In *International conference on learning representations* (pp. 1–13).
Zhang, Y., Xiang, T., Hospedales, T. M., & Lu, H. (2018). Deep mutual learning. In *Proceedings of the IEEE conference on computer vision and pattern recognition* (pp. 4320–4328).
Zhou, B., Khosla, A., Lapedriza, A., Oliva, A., & Torralba, A. (2016). Learning deep features for discriminative localization. In *Proceedings of the IEEE conference on computer vision and pattern recognition* (pp. 2921–2929).
Zhu, H., Liu, H., Zhu, C., Deng, Z., & Sun, X. (2020). Learning spatial-temporal deformable networks for unconstrained face alignment and tracking in videos. *Pattern Recognition*, *107*, Article 107354.