KICS The Korean Institute of Communications and Information Sciences

Available online at www.sciencedirect.com

ScienceDirect

ICT Express 9 (2023) 909–914

www.elsevier.com/locate/icte

Meta-ensemble learning with a multi-headed model for few-shot problems

Seokhyeon Ha, Youngseok Yoon, Jungwoo Lee*

Department of Electrical and Computer Engineering, Seoul National University, Seoul, Republic of Korea

Received 17 July 2022; accepted 1 September 2022
Available online 21 September 2022

Abstract

Recent meta-learning algorithms for few-shot learning are based on *episodic training* where each episode consists of only a few support and query samples to imitate a target few-shot task. However, due to the limited number of categories and few samples in each category, this framework suffers from over-fitting to both a meta-training dataset and the support set of each episode. It also causes a large variance in the accuracy of each episode, which reduces reliability and confidence in model performance. To address this problem, we propose a novel meta-ensemble learning approach based on a recent ensemble method: a multi-input multi-output (MIMO) configuration. Our approach is simply applied to existing meta-learning algorithms. Multiple subnetworks in a single model simultaneously learn multiple episodes and ensemble the predictions, leveraging the model capacity. We show that meta-ensemble learning achieves significant improvement in generalization. It also improves the performance of meta-learning algorithms on few-shot classification benchmarks.

Keywords: Computer vision; Few-shot learning; Meta learning

1. Introduction

Deep learning has demonstrated tremendous performance on various domains and applications. For example, deep convolutional neural networks (CNNs) have exhibited impressive performance in a wide range of vision fields, such as image recognition, object detection, and semantic segmentation. However, requiring huge amount of data samples restricts the utility of deep learning on many applications where data collection or labeling is expensive. For sample efficiency, the study of few-shot learning aims to build the deep-learning model that can generalize well using only few samples.

Meta-learning is one of the most common approaches for few-shot learning problems and has shown promising results [1–3]. It imitates human ability to learn new concepts from only a small number of samples and generalize to unseen tasks. Specifically, meta-knowledge is learned across multiple training tasks, and used to generate task-specific prior for unseen task. General meta-learning procedure is based on *episodic training* [4], where each episode (or task) consists of a support set and a query set to simulate a test environment. For image classification in this framework, classes are first randomly chosen from a finite set of categories. Then, the support set and the query set are constructed by sampling examples from these classes. In the case of few-shot problems, the support set consists of only a few examples.

Due to the small number of samples for each task and the hierarchical structure of meta-learning procedures, meta-learning algorithms for few-shot learning suffer from over-fitting to meta-train tasks or the support set of a meta-test set. The over-fitting problem is challenging, and the improvement of meta-generalization is still an open problem of meta-learning algorithms. Our key contributions are summarized as follows.

- To address generalization in meta-learning, we propose meta-ensemble learning approach combining a multi-input multi-output (MIMO) configuration with meta-learning algorithms [5].
- On various few-shot image classification benchmarks, we show that the subnetworks trained in a single model converge to distinct local minima and generate independent predictions.
- With intensive experiments, we demonstrate that our meta-ensemble learning helps to improve generalization and performance in meta-learning.

* Corresponding author.
E-mail addresses: hash1108@cml.snu.ac.kr (S. Ha), youngseok8@cml.snu.ac.kr (Y. Yoon), junglee@snu.ac.kr (J. Lee).
Peer review under responsibility of The Korean Institute of Communications and Information Sciences (KICS).

https://doi.org/10.1016/j.icte.2022.09.001

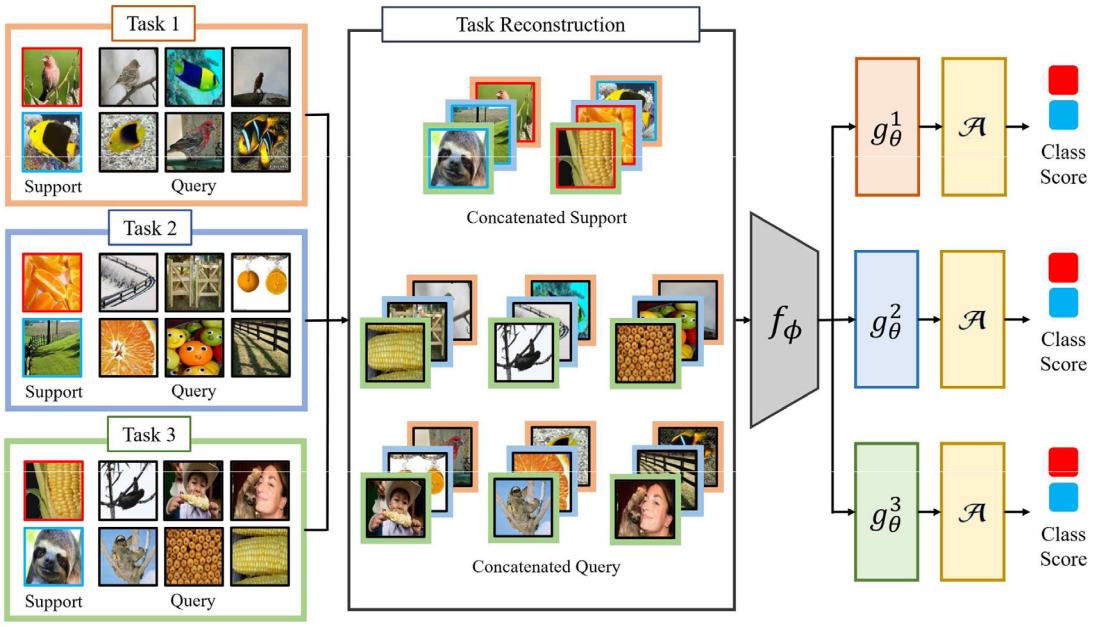


Fig. 1. Illustrative example of our meta-ensemble learning, using an $M = 3$ ensemble for a 2-way 1-shot classification problem. In each task, a support set has 2 samples from different class and a query set has 6 samples. For each training step, we get M tasks. Samples from M support sets and M query sets are randomly shuffled and concatenated to reconstruct 2 support sets and 6 query sets. There are M embedding spaces generated from feature extractor f_ϕ and M-headed network g_θ. The base-learner $\mathcal{A}$ performs M times, once for each of M feature embedding spaces.

2. Method

2.1. Meta-learning framework

Before we introduce meta-ensemble learning, we first explain meta-learning framework and define notation in this section. Most meta-learning algorithms follow an episodic training framework to train and evaluate their model. While some approaches modify the standard episodic training framework by augmenting training ways or training shots, etc. [1,2], we strictly follow the standard setting to evaluate the effect of the MIMO configuration in meta-learning only.

The entire set of categories is divided into meta-train category set $\mathcal{C}_{train}$, meta-validation category set $\mathcal{C}_{valid}$, and meta-test category set $\mathcal{C}_{test}$. To avoid memorizing problem, each category set does not overlap each other, *i.e.* , $\mathcal{C}_{train} \cap \mathcal{C}_{valid} = \emptyset$, $\mathcal{C}_{train} \cap \mathcal{C}_{test} = \emptyset$, and $\mathcal{C}_{valid} \cap \mathcal{C}_{test} = \emptyset$.

Meta-train set, meta-validation set, and meta-test set are sets of N-way K-shot tasks that consist of N classes sampled from $\mathcal{C}_{train}$, $\mathcal{C}_{valid}$, and $\mathcal{C}_{test}$. With this setup, meta-learning algorithms use meta-train sets and meta-valid sets for a meta-training stage, and use meta-test sets for a meta-test stage. For each task, a meta-train set, a meta-valid set, and a meta-test set are constructed in the same way.

To construct an N-way K-shot task $\mathcal{T}_i$, N classes are sampled from corresponding category set ($\mathcal{C}_{train}$, $\mathcal{C}_{valid}$, or $\mathcal{C}_{test}$). Then $\mathcal{T}_i = \{\mathcal{D}_i^s, \mathcal{D}_i^q\}$ is constructed by sampling a support set $\mathcal{D}_i^s$ and a query set $\mathcal{D}_i^q$. The support set $\mathcal{D}_i^s$ consists of N classes and K_s samples from each class, and the query set $\mathcal{D}_i^q$ consists of K_q samples from each class as follows:

$$\begin{aligned} \mathcal{D}_i^s &= \{(\boldsymbol{x}_i^{(n)}, y_i^{(n)}) \mid n = 1, \ldots, N \times K_s\}, \\ \mathcal{D}_i^q &= \{(\boldsymbol{x}_i^{(n)}, y_i^{(n)}) \mid n = 1, \ldots, N \times K_q\}, \end{aligned} \tag{1}$$

where $K_s = K$ for K-shot task. Note that $\mathcal{D}_i^s \cap \mathcal{D}_i^q = \emptyset$ for every task.

During a meta-training stage, meta-learning algorithms learn how to learn few-shot tasks under this framework. After that, they are evaluated in a meta-test stage.

2.2. Task level MIMO configuration

The multi-input multi-output (MIMO) configuration creates multiple subnetworks in a single model architecture by increasing the input and output size only while sharing the model body [5]. In each training step, a training batch is shuffled and concatenated multiple times to construct multiple input structures. On the other hand, we combine several different tasks into an input structures multiple times as shown in Fig. 1. A single model then learns these concatenated multiple tasks simultaneously. Therefore, by learning multiple tasks simultaneously rather than learning a single task, more generalized knowledge can be learned, which reduces over-fitting.

Specifically, we increase the number of input channels by M times to create M subnetworks in a single feature extractor f_ϕ. M images $(x_1, \ldots, x_M)$ are concatenated along the channel dimension to be used as the input $x_{1:M}$ of the feature extractor f_ϕ. The feature extractor f_ϕ generates D-dimensional feature embedding vector $f_\phi(x_{1:M}) \in \mathbb{R}^D$. For a multi-output configuration, we use a multi-headed network $g_\theta : \mathbb{R}^D \to \mathbb{R}^{M \times D}$, where mth head is denoted by $g_\theta^m : \mathbb{R}^D \to \mathbb{R}^D$. Each head of multi-headed network g_θ generates D-dimensional feature embedding vector given the feature embedding vector $f_\phi(x_{1:M})$.

In each step of meta-training, M distinct tasks $\{\mathcal{T}_m\}_{m=1}^M$ are sampled first. The detailed process of constructing each

episode is described in 2.1. The images in M episodes are concatenated along the channel dimension, and the concatenated support set $\mathcal{D}^s_{1:M}$ and the concatenated query set $\mathcal{D}^q_{1:M}$ are reconstructed as follows:

$$\begin{aligned}\mathcal{D}^s_{1:M} &= \left\{\left(\boldsymbol{x}^{(n)}_{1:M}, y^{(n)}_{1:M}\right) \mid n = 1, \ldots, N \times K_s\right\},\\ \mathcal{D}^q_{1:M} &= \left\{\left(\boldsymbol{x}^{(n)}_{1:M}, y^{(n)}_{1:M}\right) \mid n = 1, \ldots, N \times K_q\right\}.\end{aligned} \tag{2}$$

The order of M tasks to be concatenated is randomly arranged, and the order of the images in each task is also arranged randomly. Using $\mathcal{D}^s_{1:M}$ and $\mathcal{D}^q_{1:M}$, the feature extractor f_ϕ and the mth head of network g_θ generate embedding vectors $\boldsymbol{r}^{(i)}_m = g^m_\theta \circ f_\phi(\boldsymbol{x}^{(i)}_{1:M})$.

Using that, we can get the support embedding set $\mathcal{R}^s_m$ and the query embedding set $\mathcal{R}^q_m$ for the mth episode as

$$\begin{aligned}\mathcal{R}^s_m &= \{\boldsymbol{r}^{(n)}_m = g^m_\theta \circ f_\phi(\boldsymbol{x}^{(n)}_{1:M}) \mid \boldsymbol{x}^{(n)}_{1:M} \in \mathcal{D}^s_{1:M}\},\\ \mathcal{R}^q_m &= \{\boldsymbol{r}^{(n)}_m = g^m_\theta \circ f_\phi(\boldsymbol{x}^{(n)}_{1:M}) \mid \boldsymbol{x}^{(n)}_{1:M} \in \mathcal{D}^q_{1:M}\},\end{aligned} \tag{3}$$

where $m = 1, \ldots, M$.

We make prediction $\hat{y}^{(i)}_m$ on each query data $x^{(i)}_m \in \mathcal{D}^q_m$ as

$$\hat{y}^{(i)}_m = \mathcal{A}(\boldsymbol{r}^{(i)}_m; \mathcal{R}^s_m), \tag{4}$$

where $m = 1, \ldots, M$ and $\boldsymbol{r}^{(i)}_m \in \mathcal{R}^q_m$ is the mth head embedding vector of concatenated query data $x^{(i)}_{1:M}$. $\mathcal{A}(\cdot; \mathcal{R}^s_m) : \mathbb{R}^D \to \mathbb{R}^N$ represents an N-way base-learner determined by a meta-learning algorithm and a support embedding set $\mathcal{R}^s_m$. During meta-training, we train the feature extractor f_ϕ and the multi-headed network g_θ as

$$\min_{\phi,\theta} \sum_{m=1}^{M} \frac{1}{N \times K_q} \sum_{n=1}^{N \times K_q} \ell\left(\hat{y}^{(n)}_m, y^{(n)}_m\right), \tag{5}$$

where $\ell(\cdot, \cdot)$ is the cross-entropy loss.

During meta-test stage, we also reconstruct the support set $\mathcal{D}^s_{test}$ and the query set $\mathcal{D}^q_{test}$ by concatenating the same image M times along the channel dimension for meta-test task $\mathcal{T}_{test} = \{\mathcal{D}^s_{test}, \mathcal{D}^q_{test}\}$. The process of generating embedding vectors for each query input is the same as in meta-training since the same images are concatenated. The prediction for each query input $x^{(i)}_{test}$ are computed by averaging the predictions of subnetworks:

$$\hat{y}^{(i)}_{test} = \frac{1}{M} \sum_{m=1}^{M} \mathcal{A}(g^m_\theta \circ f_\phi(\boldsymbol{x}^{(i)}_{test}, \ldots, \boldsymbol{x}^{(i)}_{test}); \mathcal{R}^s_{test}). \tag{6}$$

3. Experiments

3.1. Experimental setup

Dataset. We evaluate the MIMO configuration on three few-shot image classification benchmarks: CIFAR-FS [3], FC100 [6], and miniImageNet [4]. CIFAR-FS and FC-100 are derived from CIFAR-100 [7]. CIFAR-FS consists of 64 classes for meta-training, 16 classes for meta-validation, and 20 classes for meta-test. FC-100 consists of 60 classes for meta-training, 20 classes for meta-validation, and 20 classes for meta-test. Each class in both CIFAR-FS and FC-100 contains 600 images of size 32×32. miniImageNet is derived from ImageNet [8]. It consists of 64 classes for meta-training, 16 classes for meta-validation, and 20 classes for meta-test [9]. Each class contains 600 images of size 84×84.

Base-Learner $\mathcal{A}$. We choose three representatives of metric-based meta-learning to apply our meta-ensemble learning: ProtoNet [1], R2-D2 [3], and MetaOptNet [2]. The difference between these algorithms is how to design the base-learner head $\mathcal{A}$, which is the nearest neighbor classifier using Euclidean distance, the ridge regression solver, and the SVM solver, respectively.

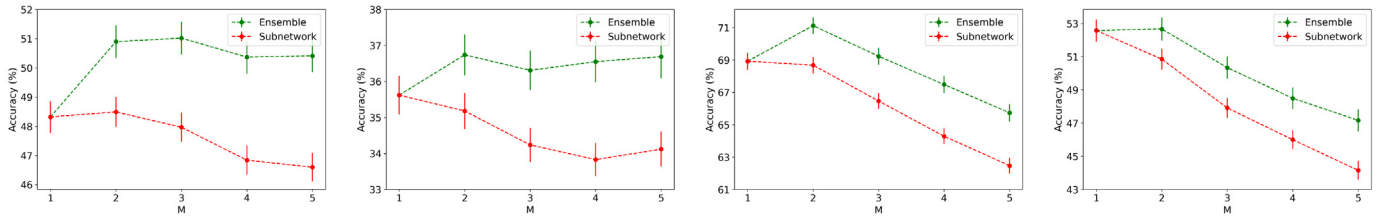
Episodic Training Framework. We conduct experiments under the 5-way 1-shot and 5-way 5-shot settings. We strictly follow the standard episodic training framework for all meta-learning algorithms. All episodes of meta-training, meta-validation, and meta-test consist of an 5way K_s-shot support set and an 5-way K_q-shot query set. Thus, We set $K_s = 1$, $K_q = 15$ and $K_s = 5$, $K_q = 15$ for all 5-way 1-shot and 5-way 5-shot tasks, respectively. We set 60 epochs for meta-training, and each epoch consists of 8000 episodes. We use 1000 episodes for meta-validation and meta-test.

Implementation Details. We use two architectures for the backbone of feature extractor f_ϕ in this paper: Conv-4 and ResNet-12. The Conv-4 architecture is composed of four 64-channel convolutional blocks [4]. The network architecture for g_θ is the following sequence: a linear layer, batch normalization, and a leaky-ReLU. To reduce the burden of computational cost and memory consumption of a linear layer that increase with the size of M, we use an average pooling at the end of feature extractor f_ϕ. It reduces the size of feature embedding to 64 and 640 for Conv-4 and ResNet-12, respectively. We use the standard data pre-processing scheme, a sequence of zero-padding with 4 pixels on each size, random crop, and horizontal flip. We use the stochastic gradient descent (SGD) optimizer with the initial learning rate of 0.1, the weight decay of 5e-4, and the Nesterov momentum of 0.9. The learning rate is decayed to 0.01, 0.001, 0.0001 at epochs 20, 40, and 50, respectively.

3.2. Performance as M varies

The number of subnetworks M is the most important hyperparameter when we construct the MIMO configuration. We conduct experiments to test how the performance of ProtoNet changes on both FC100 and miniImageNet datasets as M varies from one to five. We use the Conv-4 as backbone of the feature extractor for the experiments, as its restricted model capacity demonstrates more drastic effect on varying number of subnetworks. We also compare how the performance differs between ensemble and individual subnetworks.

Fig. 2 shows how the performance of meta-learning algorithm (ProtoNet) changes as the number of subnetworks in a single model varies. In all experiments, the performance of ProtoNet when two subnetworks are trained exceeds that of ProtoNet when a single subnetwork is trained. The performance does not degrade as M increases from two to five on the FC100 dataset, while it drops by 5 to 6% on miniImageNet which consists of higher resolution images than FC100 dataset.



(a) FC100 5-shot (b) FC100 1-shot (c) miniImageNet 5-shot (d) miniImageNet 1-shot

Fig. 2. Test accuracy (%) with 95% confidence intervals on FC100 and miniImageNet as the number of subnetworks (M) varies. The base-learner ($\mathcal{A}$) is ProtoNet, and the backbone of feature extractor (f_ϕ) is Conv-4 [4]. (a)–(b) 5-way 5-shot/1-shot learning on FC100. (c)–(d) 5-way 5-shot/1-shot learning on miniImageNet.

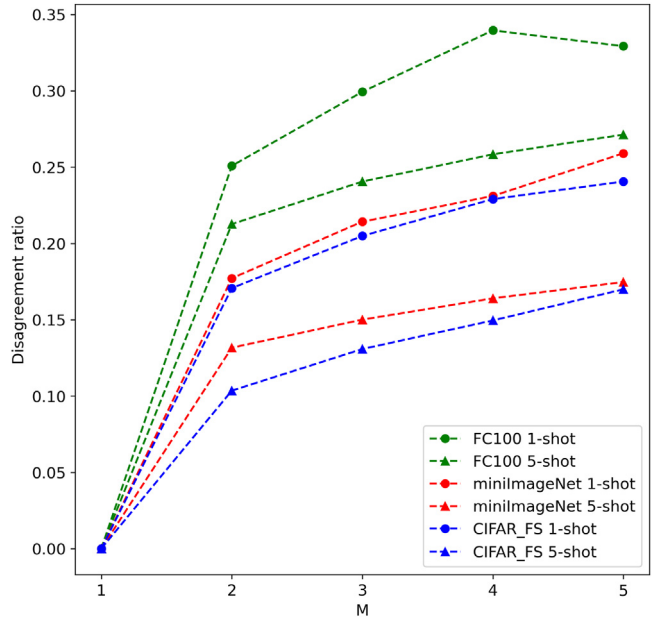


Fig. 3. Disagreement ratio as the number of subnetworks varies under the 5-way 1-shot/5-shot settings on CIFAR-FS, FC100 and miniImageNet. The base-learner ($\mathcal{A}$) is ProtoNet, and the backbone of feature extractor (f_ϕ) is Conv-4 [4].

It shows that the optimal number of subnetworks that ensemble in a single model is conditioned by the task complexity and the model capacity. Test accuracy of ensemble exceeds average test accuracy of individual subnetworks when multiple subnetworks are trained. The performance gap between ensemble and individual subnetworks increases as the number of subnetworks M increases.

3.3. Independent subnetworks

Since the ensemble effect is maximized when the ensemble members are diverse, many studies on ensemble learning focus on diversity of ensemble members. General ensemble methods use multiple networks that are trained independently [10, 11]. In those cases, difference between ensemble members is mainly originated from randomness in mini-batch training, parameter initialization, and other model-inherent arbitrariness. The parameters of these ensemble members have totally different values although they can produce similar performance in average.

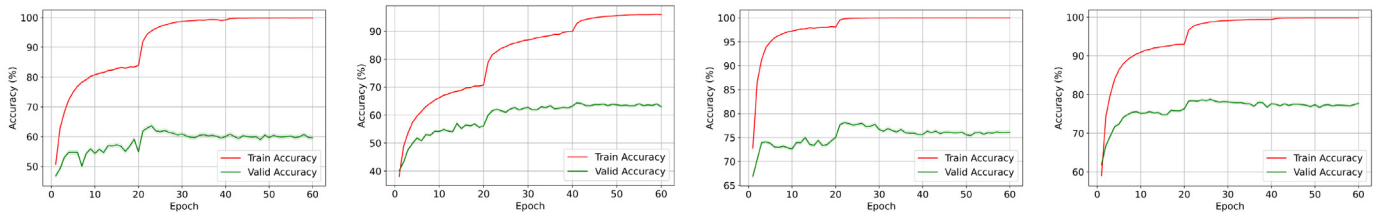
In our case, the subnetworks share a main body, but there is the independence between the subnetworks because they concurrently learn different episodes. To examine the independence between subnetworks, we measure the disagreement ratio between predictions of individual subnetworks and ensemble predictions as M varies. More specifically, for each query sample from 1000 test episodes, we average the proportion of subnetworks that predict differently from the ensemble predictions. The minimum value for the disagreement ratio is zero, when we use a single subnetwork ($M = 1$) or all subnetworks produce identical predictions. The maximum value is one, when the ensemble predictions by averaging the outputs of the subnetworks do not match any output predictions of the subnetworks for all inputs. Thus, a higher disagreement ratio indicates a more diverse distribution of outputs generated by the individual subnetworks.

In Fig. 3, we plot the disagreement ratio as the number of subnetworks M varies on three datasets. It is observed that the disagreement ratio increases as more subnetworks are trained in a single model. The disagreement ratio becomes larger when the target few-shot task is harder (1-shot rather than 5-shot, FC100 rather than miniImageNet, and miniImageNet rather than CIFAR-FS). This tendency demonstrates that the ensemble members in a single model are trained independently, producing different predictions for the same input.

3.4. Generalization improvement

Due to weak generalization of meta-learning, there is a large performance gap between meta-training and meta-test. There is a large variance in the accuracy of each episode. If the generalization ability in meta-learning is improved, the performance gap between meta-training and meta-test will narrow and the variance of accuracy across different test tasks will also decrease. Therefore, we use the learning curve to measure the generalization ability of our meta-ensemble learning.

Fig. 4 is learning curves of ProtoNet with ResNet-12 backbone on miniImagenet for 5-way 1-shot and 5-way 5-shot learning. As the accuracy is averaged over M subnetworks, the train accuracy of ensembles ($M = 2$) is generally lower than train accuracy of non-ensembles. On the other hand, the validation accuracy of ensembles is generally higher than validation accuracy of non-ensembles, and also fluctuates less with each epoch than the validation accuracy of non-ensemble. Thus, when using the ensemble in both 1-shot and 5-shot



(a) 5-way 1-shot (M=1) (b) 5-way 1-shot (M=2) (c) 5-way 5-shot (M=1) (d) 5-way 5-shot (M=2)

Fig. 4. Visualization of the generalization ability. Training and validation accuracy for ProtoNet with ResNet-12 backbone on miniImagenet. (a)–(b) 5-way 1-shot learning for baseline (M=1), and MIMO (M=2). (c)–(d) 5-way 5-shot learning for baseline (M=1), and MIMO (M=2).

Table 1
Test accuracy (%) with 95% confidence intervals on CIFAR-FS, FC100, and mini-ImageNet under 5-way 1-shot and 5-way 5-shot settings. "Conv-4" denotes the 4-layer CNN backbone [4]. "ResNet-12" denotes the ResNet-12 backbone with DropBlock regularization [2].

Method	Backbone	CIFAR-FS		FC100		mini-ImageNet	
		1-shot	5-shot	1-shot	5-shot	1-shot	5-shot
ProtoNet	Conv-4	62.83 ± 0.73	78.53 ± 0.54	35.62 ± 0.54	48.32 ± 0.54	52.58 ± 0.67	68.92 ± 0.52
+MIMO	Conv-4	61.69 ± 0.74	79.05 ± 0.53	36.74 ± 0.57	50.90 ± 0.56	52.67 ± 0.69	71.12 ± 0.52
ProtoNet	ResNet-12	69.11 ± 0.77	81.35 ± 0.53	37.11 ± 0.56	50.29 ± 0.55	58.19 ± 0.69	72.39 ± 0.53
+MIMO	ResNet-12	69.91 ± 0.76	83.18 ± 0.51	37.47 ± 0.58	51.82 ± 0.55	58.09 ± 0.70	72.73 ± 0.56
R2-D2	Conv-4	62.43 ± 0.77	78.48 ± 0.53	35.96 ± 0.56	48.08 ± 0.53	52.92 ± 0.66	68.39 ± 0.53
+MIMO	Conv-4	62.09 ± 0.76	78.82 ± 0.53	37.43 ± 0.57	51.01 ± 0.55	52.83 ± 0.66	70.27 ± 0.52
R2-D2	ResNet-12	68.55 ± 0.77	81.59 ± 0.54	37.19 ± 0.54	52.37 ± 0.57	57.14 ± 0.66	74.12 ± 0.52
+MIMO	ResNet-12	70.39 ± 0.74	84.09 ± 0.49	38.74 ± 0.58	53.93 ± 0.53	58.67 ± 0.68	75.48 ± 0.51
MetaOptNet	Conv-4	62.86 ± 0.76	77.66 ± 0.54	35.83 ± 0.53	47.70 ± 0.53	53.35 ± 0.65	67.61 ± 0.54
+MIMO	Conv-4	61.47 ± 0.75	77.81 ± 0.56	37.42 ± 0.56	49.86 ± 0.55	52.73 ± 0.65	69.08 ± 0.52
MetaOptNet	ResNet-12	69.99 ± 0.73	83.71 ± 0.48	38.12 ± 0.56	51.27 ± 0.52	57.97 ± 0.68	73.21 ± 0.51
+MIMO	ResNet-12	70.52 ± 0.73	84.14 ± 0.49	39.16 ± 0.56	52.36 ± 0.52	58.58 ± 0.66	75.10 ± 0.50

training, the gap between training and validation accuracy and the variance of accuracy across different tasks can be reduced.

3.5. Performance improvement

In this section, we demonstrate the performance improvement of three different meta-learning algorithms: ProtoNet [1], R2-D2 [3], and MetaOptNet [2]. To evaluate only the effectiveness of our meta-ensemble learning, we report our baseline results trained in our learning framework ($M = 1$). We compare these baseline performance results to our meta-ensemble learning under the 5-way 1-shot/5-shot settings on CIFAR-FS, FC100, and miniImageNet.

Table 1 shows the results of our experiments on CIFAR-FS, FC100, and mini-ImageNet. We report the average accuracies (%, top-1) with the 95(%) confidence intervals over 1000 test episodes. We were able to improve performance in most cases. In general, our MIMO configuration shows that the usage of Resnet has a greater effect than the usage of Conv4, indicating that it works more effectively in more complex network structures. For the number of subnetworks in the MIMO configuration, we choose the optimal M for each dataset and backbone. We set $M = 4$ for ResNet-12 on FC100 and $M = 2$ for all other experiments.

4. Conclusion

Weak generalization and over-fitting are the main problems that meta-learning for few-shot problems needs to tackle with. We propose a novel ensemble learning approach for meta-learning algorithms that can be applied with other generalization methods. We design a MIMO configuration based method suitable for few-shot image classification tasks, which shows that the ensemble of multiple subnetworks does not degrade the performance of meta-learning algorithms. It is also shown that it improves generalization and performance in meta-learning.

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 the Agency For Defense Development by the Korean Government (UD190031RD (50%)), Institute of Information & communications Technology Planning & Evaluation (IITP, 2021-0-00106 (30%)), National R&D Program through the National Research Foundation of Korea (NRF) funded by Ministry of Science and ICT (2021M3F3A2A02037893 (20%)), INMAC, and BK21-plus.

References

[1] J. Snell, K. Swersky, R. Zemel, Prototypical networks for few-shot learning, Adv. Neural Inf. Process. Syst. 30 (2017) 4077–4087.

[2] K. Lee, S. Maji, A. Ravichandran, S. Soatto, Meta-learning with differentiable convex optimization, in: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, 2019, pp. 10657–10665.
[3] L. Bertinetto, J.F. Henriques, P. Torr, A. Vedaldi, Meta-learning with differentiable closed-form solvers, in: International Conference on Learning Representations, 2018.
[4] O. Vinyals, C. Blundell, T. Lillicrap, D. Wierstra, et al., Matching networks for one shot learning, Adv. Neural Inf. Process. Syst. 29 (2016) 3630–3638.
[5] M. Havasi, R. Jenatton, S. Fort, J.Z. Liu, J. Snoek, B. Lakshminarayanan, A.M. Dai, D. Tran, Training independent subnetworks for robust prediction, in: International Conference on Learning Representations, 2020.
[6] B.N. Oreshkin, P.R. López, A. Lacoste, TADAM: Task dependent adaptive metric for improved few-shot learning, in: NeurIPS, 2018.
[7] A. Krizhevsky, et al., Learning Multiple Layers of Features from Tiny Images, Citeseer, 2009.
[8] O. Russakovsky, J. Deng, H. Su, J. Krause, S. Satheesh, S. Ma, Z. Huang, A. Karpathy, A. Khosla, M. Bernstein, et al., Imagenet large scale visual recognition challenge, Int. J. Comput. Vis. 115 (3) (2015) 211–252.
[9] S. Ravi, H. Larochelle, Optimization as a model for few-shot learning, in: ICLR, 2017.
[10] S. Lee, S.P.S. Prakash, M. Cogswell, V. Ranjan, D. Crandall, D. Batra, Stochastic multiple choice learning for training diverse deep ensembles, in: Advances in Neural Information Processing Systems, 2016, pp. 2119–2127.
[11] B. Lakshminarayanan, A. Pritzel, C. Blundell, Simple and scalable predictive uncertainty estimation using deep ensembles, Adv. Neural Inf. Process. Syst. 30 (2017).