A Nested Bi-level Optimization Framework for Robust Few Shot Learning

Krishnateja Killamsetty*, Changbin Li*, Chen Zhao, Feng Chen, Rishabh Iyer

The University of Texas at Dallas, Richardson, Texas, USA
{krishnateja.killamsetty, changbin.li, chen.zhao, feng.chen, rishabh.iyer}@utdallas.edu

Abstract

Model-Agnostic Meta-Learning (MAML), a popular gradient-based meta-learning framework, assumes that the contribution of each task or instance to the meta-learner is equal. Hence, it fails to address the domain shift between base and novel classes in few-shot learning. In this work, we propose a novel robust meta-learning algorithm, NESTEDMAML, which learns to assign weights to training tasks or instances. We consider weights as hyper-parameters and iteratively optimize them using a small set of validation tasks set in a nested bi-level optimization approach (in contrast to the standard bi-level optimization in MAML). We then apply NESTED-MAML in the meta-training stage, which involves (1) several tasks sampled from a distribution different from the meta-test task distribution, or (2) some data samples with noisy labels. Extensive experiments on synthetic and real-world datasets demonstrate that NESTEDMAML efficiently mitigates the effects of "unwanted" tasks or instances, leading to significant improvement over the state-of-the-art robust meta-learning methods.

Introduction

Meta-learning (Schmidhuber 1987; Naik and Mammone 1992; Santoro et al. 2016; Vinyals et al. 2016; Finn, Abbeel, and Levine 2017) can achieve quick adaption for UNSEEN tasks by identifying common structures among various SEEN tasks, enabling faster learning of a new task with as little data as possible. However, existing meta-learning techniques (*e.g.,* MAML (Finn, Abbeel, and Levine 2017)) often fail to generalize well when the test tasks belong to a different distribution from the training tasks distribution (Chen et al. 2019). For example, MAML assumes equal weights to all samples and tasks during meta-training. This task homogeneity assumption of MAML often limits its ability to work in real-world applications (Wei and Kehtarnavaz 2020).

We motivate the importance of robust meta-learning when meta-training tasks have OOD tasks using the following examples. For example, consider the task of detecting vehicles at night under different weather conditions. In this case, the meta-test tasks only consist of images of vehicles at night. Since the procurement vehicles driving data at night, covering all critical scenarios is difficult, we need a model that can

*These authors contributed equally.

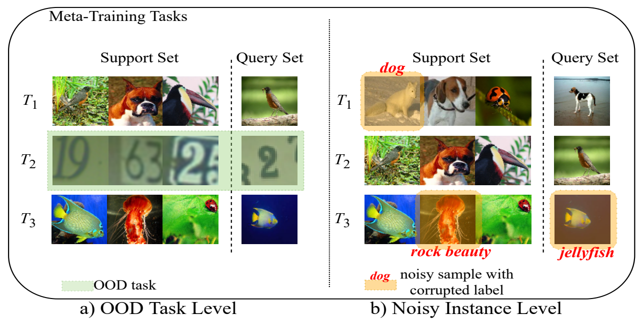


Figure 1: We consider corrupted training set for few-shot learning in this work: a) OOD task level and b) noisy instance level. T_2 in a) is an OOD task that is sampled from a different distribution. b) contains some noisy samples which are mislabeled. For example, the actual label of the first sample in T_1 should be "*Arctic fox*" which is labeled as "*dog*"; The labels of two noisy samples in T_3 are flipped wrongly. The first one should be "*rock beauty*," and the other one should be "*jellyfish*."

quickly adapt to rare driving conditions. Hence, we consider meta-training tasks to consist of images of the vehicles in multiple lighting scenarios. In this case, some of the tasks in meta-training may degrade the meta-test performance. So, it is vital to have a meta-learning model that is robust to OODs.

In the examples given above, meta-test tasks belong to specialized slices where the data availability is meager compared to meta-training tasks. The meta-training task distribution is biased compared to that in the meta-test. To keep the whole meta-training tasks for generalization and reduce the adverse impact of the biased distribution in meta-training, we propose a novel robust few-shot learning algorithm in the presence of outliers in meta-training time, which is similar to the corruptions in training time in the traditional robust learning (Schneider et al. 2020). This is different from the existing robust few-shot learning papers (Yin et al. 2018; Lu et al. 2020; Goldblum, Fowl, and Goldstein 2020) which consider the corruption only happens in meta-test time.

To simulate the corruptions in meta-training, two levels of outliers (Figure 1) are considered: *a*) Out-Of-Distribution (OOD) task, where the meta-training has tasks that are out of

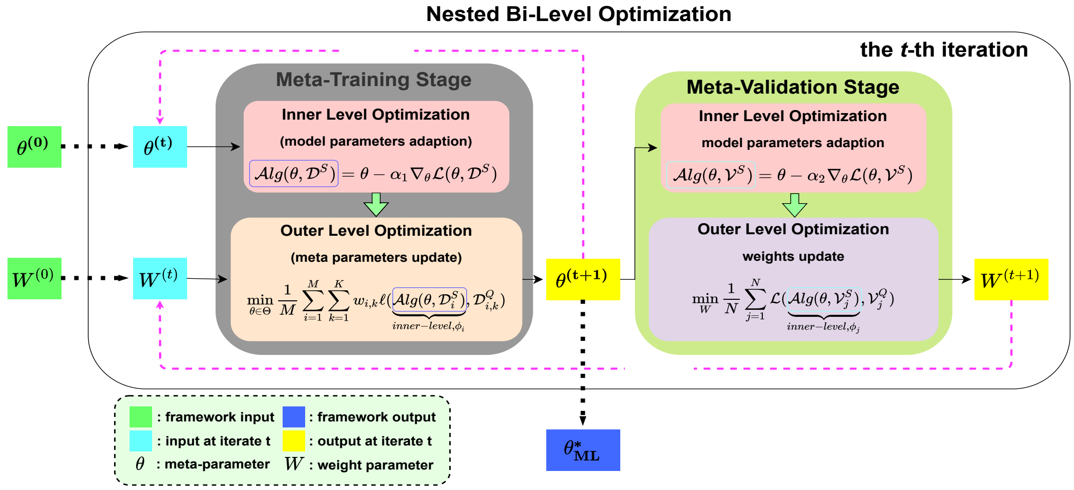


Figure 2: Overview of our NestedMAML framework that solves a *nested bi-level* optimization problem. (a) In the meta-training stage, model parameters ϕ_i of each task are adapted from meta-parameter $\boldsymbol{\theta}$ through the inner level optimization; (b) In the outer-level of the meta-training stage, we update the meta-parameters using the weights W from the previous iterate; (c) Weights are further updated in the meta-validation stage using the gradient of the meta-losses with respect to current W.

distribution to the meta-test tasks (i.e., the meta-test dataset is a specialized slice of meta-train) and *b*) noisy instance level, where some of the labels might be noisy (due to human labeling errors or inherent ambiguity of certain classification problems) for meta-training samples.

A natural way of dealing with corrupted data in meta-training is by assigning weights to either tasks or individual instances. For example, assigning zero weight to OOD or noisy tasks/instances in the meta-train set improves the meta-learning algorithm's performance. Inspired by (Ren et al. 2018), in this work, we propose an end-to-end robust meta-learning framework called NestedMAML that can achieve the reweighting schema along with learning good model initialization parameters in the few-shot learning scenario.

NestedMAML considers the weights as hyper-parameters and uses a small set of meta-validation tasks representing the meta-test tasks to find the optimal hyper-parameters by minimizing the meta-loss on the validation tasks in a **nested bi-level** manner. An overview of NestedMAML is given in Figure 2. In practice, the size of the meta-validation tasks set required by NestedMAML is tiny compared to the meta-training dataset. Hence, creating a small and clean meta-validation set is neither expensive nor unrealistic, even for rare specialized use cases of a real-life scenario. A similar strategy has been applied in (Ren et al. 2018; Shu et al. 2019; Killamsetty et al. 2020). However, they focus on traditional supervised learning, and we generalize this to task- and instance-level in a meta-learning setting. Since NestedMAML uses an online framework to perform a joint optimization of the weight hyper-parameters and model parameters for the weighted MAML model, the computational time of ours is comparable to MAML.

Contributions of our work are summarized as follows: 1) We study the general form of the task and instance weighted meta-learning, where we learn the optimal weights and model initialization parameters by optimizing a *nested bi-level* objective function. To the best of our knowledge, ours is the first work that studies the *nested bi-level* optimization problem, which comes naturally in such a new setting. 2) We introduce a novel algorithmic framework NestedMAML that uses a small set of validation tasks to enable robust meta-learning. We solve the *nested bi-level* optimization problem efficiently through a series of practical approximations and provide a theoretical convergence analysis for NestedMAML. In particular, we show that NestedMAML converges in $\mathcal{O}(1/\epsilon^2)$ iterations under reasonable assumptions and contrast this with existing bounds of MAML. 3) We provide comprehensive synthetic and real-world data experiments demonstrating that NestedMAML achieves state-of-the-art results in two scenarios (OOD tasks and noisy instance labels).

Related Work

There are several lines of meta-learning algorithms: nearest neighbors-based methods (Vinyals et al. 2016), recurrent network-based methods (Ravi and Larochelle 2016), and gradient-based methods. As the representative of gradient-based meta-learning algorithms, MAML (Finn, Abbeel, and Levine 2017) and its variants (Zhao, Chen, and Thuraisingham 2021; Zhao et al. 2020a; Finn, Xu, and Levine 2018; Nichol, Achiam, and Schulman 2018; Rusu et al. 2018; Rajeswaran et al. 2019; Behl, Baydin, and Torr 2019; Raghu et al. 2019; Zhao et al. 2020b; Zhou, Knowles, and Finn 2021) learn a shared initialization of model parameters across a variety of tasks during the meta-training phase that can adapt to new tasks using a few gradient steps. Cai et al. (2020) proposes a simple weighted meta-learning approach only for the basis regression problem that selects weights by minimizing a data-dependent bound involving an empirical integral probability metric between the weighted sources and target risks. However, this approach cannot be easily extended to complex scenarios with arbitrary loss functions.

There are few meta-learning papers discussing learning with OOD tasks. Jeong and Kim (2020) propose an OOD detection framework in meta-learning through generating fake samples which resemble in-distribution samples and combine them with real samples. However, they assume the outlier instances exist in the query set, which is different from ours. The most relevant field is from the perspective of task heterogeneity (Vuorio et al. 2019; Triantafillou et al. 2020; Yao et al. 2020). Vuorio et al. (2019) proposed MMAML to deal with multimodal task distribution with disjoint and far apart modes and generates a set of separate meta-learned prior parameters to deal with each mode of a multimodal distribution. If we view that all the OOD tasks belong to a single mode, this is relevant to our setting. To tackle the distribution drift from meta-training to meta-test, B-TAML (Lee et al. 2020) learn to relocate the initial parameters to a new start point based on the arriving unseen tasks in the meta-test. The setting considered in our work and B-TAML work can be viewed as similar if we assume some of the datasets considered in the multi-dataset classification setting of B-TAML as OOD datasets.

To tackle samples with corrupted labels, some researches (Luo et al. 2015; Jalal et al. 2017; Wang and Yu 2019) introduce noise-robust models. Ren et al. (2018) and Shu et al. (2019) propose a noisy data filtering strategy using an instance reweighting strategy where the weights are learned automatically. However, the effect of noisy labels on few-shot learning requires more attention. Although (Yin et al. 2018; Lu et al. 2020; Goldblum, Fowl, and Goldstein 2020) proposes robust meta-learning or few-shot learning, they assume a presence of outliers containing in meta-test, which is different from ours.

Preliminaries

Notations

In the setting of meta-learning for few-shot learning, there is a set of meta-training tasks $\{\mathcal{T}_i\}_{i=1}^M$ sampled from the probability distribution $p_{tr}(\mathcal{T})$. Each few-shot learning task $\mathcal{T}_i$ has an associated dataset $\mathcal{D}_i$ containing two disjoint sets $\{\mathcal{D}_i^S, \mathcal{D}_i^Q\}$, where the superscripts S and Q denote support set and query set respectively. The query sets take the form $\mathcal{D}_i^Q = \{\boldsymbol{x}_i^k, y_i^k\}_{k=1}^K$ and similarly for $\mathcal{D}_i^S$. Meta-validation tasks are denoted in a similar manner: $\{\mathcal{T}_j^{\mathcal{V}} = \{\mathcal{V}_j^S, \mathcal{V}_j^Q\}\}_{j=1}^N$ Let the loss function be denoted as $\mathcal{L}(\boldsymbol{\phi}, \mathcal{D})$ with $\boldsymbol{\phi}$ denoting model parameters and $\mathcal{D}$ denoting the dataset, and $\ell(\boldsymbol{\theta}, d)$ with model parameters $\boldsymbol{\theta}$ on the data-point d. For example, $\mathcal{L}(\phi, \mathcal{D}_i^Q)$ denotes the loss of the i^{th} training task query set $\mathcal{D}_i^Q$ for given model parameters $\phi \in \boldsymbol{\Phi} \equiv \mathbb{R}^d$, where $\boldsymbol{\phi} := \mathcal{A}lg(\boldsymbol{\theta}, \mathcal{D}^S)$ and $\boldsymbol{\theta} \in \boldsymbol{\Theta} \equiv \mathbb{R}^d$ is the meta-parameter. $\mathcal{A}lg(\cdot)$ corresponds to a learning algorithm.

For notation convenience, we write $\mathcal{L}_i(\phi) := \mathcal{L}(\phi, \mathcal{D}_i^Q)$; $\mathcal{L}_{V_j}(\phi) := \mathcal{L}(\phi, \mathcal{V}_j^Q)$; $\widehat{\mathcal{L}}_{V_j}(\boldsymbol{\phi}) := \mathcal{L}(\boldsymbol{\phi}, \mathcal{V}_j^S)$. We denote scalars by lower case italic letters, vectors by lower case boldface letters, and matrices by capital italic letters throughout the paper. A table of notations with corresponding explanations is given in Appendix A.

Model-Agnostic Meta-Learning

The goal of MAML (Finn, Abbeel, and Levine 2017) is to obtain the optimal initial parameters that minimize the meta-training objective:

$$\begin{aligned} &\overbrace{\boldsymbol{\theta}^*_{ML} = \arg\min_{\boldsymbol{\theta}\in\boldsymbol{\Theta}} \mathcal{F}(\boldsymbol{\theta})}^{outer-level} \\ &\text{where, } \mathcal{F}(\boldsymbol{\theta}) = \frac{1}{M}\sum\nolimits_{i=1}^{M} \mathcal{L}(\overbrace{\mathcal{A}lg(\boldsymbol{\theta}, \mathcal{D}_i^S)}^{inner-level}, \mathcal{D}_i^Q) \end{aligned} \quad (1)$$

This is a bi-level optimization problem, where we construe that $\mathcal{A}lg(\boldsymbol{\theta}, \mathcal{D}_i^S)$ explicitly or implicitly optimizes the inner-level task-specific adaptation. The outer-level corresponds to the meta-training objective of generalizing well (i.e. low test error) on the query set of each task after adaptation.

Since $\mathcal{A}lg(\boldsymbol{\theta}, \mathcal{D}_i^S)$ corresponds to single or multiple gradient descent steps. In case of a single gradient descent, $\mathcal{A}lg(\boldsymbol{\theta}, \mathcal{D}_i^S)$ can be perceived as follwing:

$$\mathcal{A}lg(\boldsymbol{\theta}, \mathcal{D}_i^S) = \boldsymbol{\theta} - \alpha\nabla_{\boldsymbol{\theta}}\mathcal{L}(\boldsymbol{\theta}, \mathcal{D}_i^S) \quad (2)$$

where α is a learning rate. As shown above, the meta-training objective assumes equal weights to each task for generalization, which may not be ideal in the case of adversaries in the training tasks set.

Methodology

Problem Formulation

This section discusses a more generalized meta-learning framework, where we weigh all the data instances in the query set of a task. One of the significant purposes for considering weighted meta-learning is to make it more robust to adversaries during training.

In meta-learning, the support and query datasets $\{\mathcal{D}_i^S, \mathcal{D}_i^Q\}$ for each task $\mathcal{T}_i$ are usually sampled from an underlying dataset $\mathcal{D}$. In *instance-level weighting*, we associate each data instance $\{\mathcal{D}_{ik}^Q \mid k \in [K]\}$ in the query set of task $\mathcal{T}_i$ with a particular weight w_{ik}, where K is the number of datapoints (instances) in the query set $\mathcal{D}_i^Q$. The problem can be formulated as follows:

$$\begin{aligned} &\boldsymbol{\theta}^*_{ML} = \arg\min_{\boldsymbol{\theta}\in\boldsymbol{\Theta}} \mathcal{F}_w(\boldsymbol{\theta}) \\ \text{where } \; &\mathcal{F}_w(\boldsymbol{\theta}) = \frac{1}{M}\sum\nolimits_{i=1}^{M}\sum\nolimits_{k=1}^{K} w_{ik}\ell(\mathcal{A}lg(\boldsymbol{\theta}, \mathcal{D}_i^S), \mathcal{D}_{ik}^Q) \\ &= \frac{1}{M}\sum\nolimits_{i=1}^{M} \mathbf{w}_i\mathcal{L}_i(\mathcal{A}lg(\boldsymbol{\theta}, \mathcal{D}_i^S)) \end{aligned} \quad (3)$$

In the expression above,

$$\mathcal{L}_i(\mathcal{A}lg(\boldsymbol{\theta}, \mathcal{D}_i^S)) = \begin{bmatrix} \ell(\mathcal{A}lg(\boldsymbol{\theta}, \mathcal{D}_i^S), \mathcal{D}_{i1}^Q) \\ \cdots \\ \ell(\mathcal{A}lg(\boldsymbol{\theta}, \mathcal{D}_i^S), \mathcal{D}_{ik}^Q) \\ \cdots \\ \ell(\mathcal{A}lg(\boldsymbol{\theta}, \mathcal{D}_i^S), \mathcal{D}_{iK}^Q) \end{bmatrix}$$

and $\mathbf{w}_i = [w_{i1}, \ldots, w_{iK}]$ is the weight vector corresponding to the query set of task $\mathcal{T}_i$. The *instance-level weighting* is useful in the scenarios where our underlying dataset $\mathcal{D}$

is prone to noisy labeled instances where an appropriate instance-level weighting can be used to distinguish the noisy samples with corrupted labels in the task. An ideal weight assignment is assigning large weight values to clean samples and small weight values to noisy samples in a task.

Likewise, we discuss a special case of the instance weighting scheme called *task-level weighting*, where we assign equal weights to every instance in the query set of a single task. *Task-level weighting* is applied in scenarios where every instance in a task's query set is from an OOD task distribution or an In-Distribution (ID) task. In this case, the optimal weight assignment assigns small weight values to an OOD task and large weight values to an ID task.

NESTED BI-LEVEL Optimization

Since we do not know the optimal weight assignment for real-world datasets, we need to learn the weights before training the *instance-level weighting* model using the bi-level optimization problem defined in Eq.(3).

NESTEDMAML solves for optimal weight assignments by posing them as hyper-parameters using the optimization problem defined in Eq.(4). As seen in the optimization equation, NESTEDMAML uses a clean held-out meta-validation task set $\{\mathcal{T}_j^{\mathcal{V}} = \{\mathcal{V}_j^S, \mathcal{V}_j^Q\}\}_{j=1}^N$ that is assumed to be relevant to test task distribution for generalization performance. In practice, the meta-validation task set's size is small compared to that of the meta-training tasks set ($N \ll M$). Hence, NESTEDMAML tries to select the weight hyper-parameters minimizing the model's meta-validation loss after taking a few gradient steps from the initial model parameters set using the instance-level weighting scheme.

The weight optimization objective for the instance-weighted MAML schema is as follows:

$$\begin{aligned} W^* &= \operatorname*{arg\,min}_{\mathbf{w}} \frac{1}{N} \sum\nolimits_{j=1}^{N} \mathcal{L}(\mathcal{A}lg(\boldsymbol{\theta}_W^*, \mathcal{V}_j^S), \mathcal{V}_j^Q) \\ \text{where } \boldsymbol{\theta}_W^* &= \operatorname*{arg\,min}_{\boldsymbol{\theta} \in \boldsymbol{\Theta}} \frac{1}{M} \sum\nolimits_{i=1}^{M} \mathbf{w}_i^* \mathcal{L}(\mathcal{A}lg(\boldsymbol{\theta}, \mathcal{D}_i^S), \mathcal{D}_i^Q) \end{aligned} \tag{4}$$

and $W = [\mathbf{w}_1, \ldots, \mathbf{w}_M]^\intercal$. Since the optimization problem for $\boldsymbol{\theta}_W^*$ is a standard bi-level optimization problem (*i.e.* MAML), the complete optimization problem (Eq.(4)) turns out to be a **nested bi-level** optimization problem. It involves solving a standard bi-level optimization problem for every weight configuration, and hence naively solving this **nested bi-level** optimization problem is intractable. Hence, we adopt an online and one-step meta-gradient based approach to solve the optimization problem more efficiently.

The NESTEDMAML Algorithm

To reduce the optimization problem's (Eq.(4)) computation complexity, we solve the optimization problem in an iterative manner where we optimize the model parameters and weight hyperparameter by taking a single gradient step. This process is repeated until we reach convergence. Hence, we approximate the solution to the model parameters optimization in Eq.(4) first by adapting to each task using a single gradient step towards the inner task adaptation objective's descent direction and then taking a single gradient step towards the meta objective's descent direction.

Assuming that at every iterate t of training, a mini-batch of training tasks $\{\mathcal{T}_i \mid 1 \le i \le m\}$ is sampled, where m is the mini-batch size and $m \ll M$, the optimal model parameters update of the above problem is as follows:

$$\boldsymbol{\theta}_W^{(t)} = \boldsymbol{\theta}^{(t)} - \eta \frac{1}{m} \sum\nolimits_{i=1}^{m} \mathbf{w}_i^{(t)} \nabla_{\boldsymbol{\theta}} \mathcal{L}_i(\mathcal{A}lg(\boldsymbol{\theta}, \mathcal{D}_i^S))|_{\boldsymbol{\theta}^{(t)}} \tag{5}$$

where η is meta objective's step-size and α is the inner objective's step-size. After this, the optimal weight optimization problem will be as follows:

$$W^* = \operatorname*{arg\,min}_{W} \frac{1}{N} \sum\nolimits_{j=1}^{N} \mathcal{L}_{V_j}(\mathcal{A}lg(\boldsymbol{\theta}_W^{(t)}, \mathcal{V}_j^S)) \tag{6}$$

Similarly, we optimize the weight hyperparameters by taking a single gradient step towards the meta-validation loss descent. We want to evaluate the impact of training a model on the weighted MAML objective against the meta-objective of sampled validation tasks $\{\mathcal{T}_j^V \mid 1 \le j \le n\}$ where, n is the mini-batch size and $n \ll N$. The weight update equation for the instance weighting scheme is as follows:

$$W^{(t+1)} = W^{(t)} - \frac{\gamma}{n} \sum\nolimits_{j=1}^{n} \nabla_W \mathcal{L}_{V_j}(\mathcal{A}lg(\boldsymbol{\theta}_W^{(t)}, \mathcal{V}_j^S)) \tag{7}$$

where γ is the weight update's step size. The Lemma below provides the gradient of the meta-validation loss $\frac{1}{n} \sum_{j=1}^{n} \nabla_W \mathcal{L}_{V_j}(\mathcal{A}lg(\boldsymbol{\theta}_W^{(t)}, \mathcal{V}_j^S))$ w.r.t. the weight vector $\mathbf{w}_i$, therefore giving the full update equation.

Lemma 1. *The weight update for an individual weight vector* $\mathbf{w}_i$ *of the task* $\mathcal{T}_i$ *from time step* t *to* $t+1$ *is as follows:*

$$\begin{aligned} \mathbf{w}_i^{(t+1)} = \mathbf{w}_i^{(t)} + \frac{\eta\gamma}{mn} \sum\nolimits_{j=1}^{n} \nabla_{\phi_j} \mathcal{L}_{V_j} \Big(& \nabla_{\boldsymbol{\theta}} \mathcal{L}_i(\mathcal{A}lg(\boldsymbol{\theta}, \mathcal{D}_i^S))^\intercal \\ & - \alpha \nabla^2 \widehat{\mathcal{L}}_{V_j}|_{\boldsymbol{\theta}_W^{(t)}} \nabla_{\boldsymbol{\theta}} \mathcal{L}_i(\mathcal{A}lg(\boldsymbol{\theta}, \mathcal{D}_i^S))^\intercal \Big) \end{aligned} \tag{8}$$

where $\phi_j = \mathcal{A}lg(\boldsymbol{\theta}, \mathcal{V}_j^S)$.

The proof is in Appendix B. Once the optimal weights $\mathbf{w}^{(t+1)}$ at $t+1$ are achieved, we train the model using the new weights:

$$\boldsymbol{\theta}^{(t+1)} = \boldsymbol{\theta}^{(t)} - \frac{\eta}{m} \sum\nolimits_{i=1}^{m} \mathbf{w}_i^{(t+1)} \nabla_{\boldsymbol{\theta}} \mathcal{L}_i(\mathcal{A}lg(\boldsymbol{\theta}^{(t)}, \mathcal{D}_i^S)) \tag{9}$$

We repeat the steps given in the equation (5) from $t = 1$ until convergence. See Algorithm 1 for the full pseudo-code of NESTEDMAML.

First-Order Approximation (NESTEDMAML-FO). Even after the one step gradient approximation, the weight gradient calculation involves calculating multiple Hessian vector products, which is expensive. Since the coefficient of the Hessian vector-product term in the weight update (Eq. (8)) involves the product of three learning rate terms $\eta\alpha\gamma$, we can make an approximation that the term involving the Hessian vector-product term is close to 0, given that the above learning rates are small. The approximated weight update takes the following form:

$$\mathbf{w}_i^{(t+1)} = \mathbf{w}_i^{(t)} + \frac{\eta\gamma}{mn} \sum\nolimits_{j=1}^{n} \nabla_{\phi_j} \mathcal{L}_{V_j} \nabla_{\boldsymbol{\theta}} \mathcal{L}_i(\mathcal{A}lg(\boldsymbol{\theta}, \mathcal{D}_i^S))^\intercal \tag{10}$$

This approximation is similar to the first-order approximation given in (Finn, Abbeel, and Levine 2017) where the second

Algorithm 1: NESTEDMAML

Require: p_{tr}, p_{val} distribution over training, validation tasks
Require: m, n (batch sizes) and α, η, γ (learning rates)

```
Randomly initialize θ and W
while not done do
    Sample mini-batch of tasks {D_i^S, D_i^Q}_{i=1}^m ~ p_tr
    Sample mini-batch of tasks {V_j^S, V_j^Q}_{j=1}^n ~ p_val
    for each task T_i, ∀i ∈ [1, m] do
        Compute adapted parameters Alg(θ, D_i^S) with gradient
        descent by Eq. (2)
        Compute the gradient ∇_θ L_i(Alg(θ, D_i^S)) using D_i^Q
        Formulate the θ as a function of weights θ_W^(t) by Eq. (5)
        Update w_i^(t) by Eq.(8) using {V_j^S, V_j^Q}_{j=1}^n
    end for
    Update θ^(t+1) by Eq. (9) using {D_i^Q}_{i=1}^m
end while
```

and higher-order terms are neglected. We want to show a faster way to solve the *nested bi-level* weight optimization problem with a tradeoff in performance. Our experimental results show that we achieve state-of-the-art performance using NESTEDMAML. Our results also show that NESTEDMAML-FO leads to a loss in performance with a commensurate gain in speed compared to the unmodified NESTEDMAML version.

Weights Sharing. The number of weight hyper-parameters in the instance-level weighting scheme correlates to the number of data instances in the query sets of the meta-training tasks. We need to determine a significant amount of hyperparameters if the number of training tasks or data instances is enormous, which in turn affects the hyper-parameter optimization algorithm, leading to instabilities during training. Accordingly, we seek to evaluate a smaller number of hyperparameters by sharing the weights among instances. The task-weighting scheme is an occurrence of weight sharing where we share the same weight among all the instances in the query set. Apart from the task-level weighting scheme, we try to cluster tasks based on some similarity criteria to share the same weight among all the data instances in a cluster's query sets. We likewise present a sensitive analysis in the experiment section illustrating how the number of clusters in the training tasks or instances affects the NESTEDMAML algorithm's performance.

Convergence of NESTEDMAML Algorithm

Although the MAML algorithm's convergence rate is studied (Balcan, Khodak, and Talwalkar 2019; Fallah, Mokhtari, and Ozdaglar 2020; Finn et al. 2019), those results do not directly hold in our case since we have a *nested bi-level* optimization objective instead of standard bi-level objective of the MAML. Recall that in the case of strongly convex losses, MAML admits a convergence rate of $\mathcal{O}(1/\epsilon)$ (Balcan, Khodak, and Talwalkar 2019; Finn et al. 2019). In contrast, for the non-convex case, (Fallah, Mokhtari, and Ozdaglar 2020) show a weaker convergence rate of $\mathcal{O}(1/\epsilon^2)$ to a first order stationary point. In this work, we show that NESTEDMAML achieves a convergence rate of $\mathcal{O}(1/\epsilon^2)$ in the case of convex losses, as long as the inner learning rate is not too high. Furthermore, we show that NESTEDMAML converges to a critical point of meta-validation loss and not the meta-training loss since we are optimizing the meta-validation loss in the nested bi-level setting. Table 1 shows the convergence rates of MAML and NESTEDMAML algorithms for strongly convex and non-convex loss functions.

Algorithm	Strongly Convex Loss	Non-Convex Loss
MAML	$\mathcal{O}(1/\epsilon)$	$\mathcal{O}(1/\epsilon^2)$
NESTEDMAML	$\mathcal{O}(1/\epsilon^2)$	Open

Table 1: Convergence Rates of MAML and NESTEDMAML

Theorem 1. *Suppose the loss function $\mathcal{L}(\cdot)$ is Lipschitz smooth with constant L, μ-strongly convex, and is a twice differential function with a ρ-bounded gradient and $\mathcal{B}$-Lipschitz Hessian. Denote σ as the variance of drawing uniformly mini-batch sample at random. Assume that the learning rate η_t satisfies $\eta_t = \min(1, k/T)$ for some $k > 0$ such that $k/T < 1$ and γ_t, $1 \leq t \leq T$, is a monotone descent sequence. Let $\gamma_t = \min\left(\frac{1}{L}, \frac{C}{\sigma\sqrt{T}}\right)$ for some $C > 0$ such that $\frac{\sigma\sqrt{T}}{C} \geq L$ and $\sum_{t=0}^{\infty} \gamma_t \leq \infty$, $\sum_{t=0}^{\infty} \gamma_t^2 \leq \infty$. Then,* NESTEDMAML *satisfies:* $\mathbb{E}\left[\left\|\frac{1}{N}\sum_{j=1}^{N}\frac{\partial \mathcal{L}}{\partial W}\right\|^2\right] \leq \epsilon$ *in* $\mathcal{O}(\frac{1}{\epsilon^2})$ *steps. More specifically,*

$$\min_{0 \leq t \leq T} \mathbb{E}\left[\left\|\frac{1}{N}\sum_{j=1}^{N}\frac{\partial \mathcal{L}}{\partial W}\right\|^2\right] \leq \mathcal{O}\left(\frac{1}{\sqrt{T}}\right)$$

$$\textit{where } \frac{\partial \mathcal{L}}{\partial W} = \nabla_W \mathcal{L}(\mathcal{A}lg(\boldsymbol{\theta}_W^{(t)}, \mathcal{V}_j^S), \mathcal{V}_j^Q).$$

Proof is given in Appendix C. The difference in convergence rates between MAML and NESTEDMAML is due to the additional complexity involved in solving a *nested bi-level* optimization problem. The convergence analysis of NESTEDMAML for non-convex functions is challenging and currently unknown. Even though most deep learning problems have a non-convex landscape, the algorithms initially developed for convex cases have shown promising empirical results in non-convex cases. Under this assumption, we provide an implementation that can be generalized to any deep learning architecture in Algorithm 1.

Experiments

In order to corroborate NESTEDMAML, we aim to study two questions: **Q1**: Can NESTEDMAML be successfully applied to problems where task distribution in the training domain is partially shifted from the task distribution in the testing domain? **Q2**: Instead of learning task weights, can NESTEDMAML deal with problems where data instances with noisy labels are used during the meta-training stage by learning weights in an instance-level scheme?

To answer these questions, we conduct the following experiments: (1) Mix OOD tasks with the meta-training tasks to evaluate the task-level weighting scheme of NESTEDMAML and (2) corrupt the labels of some training samples to evaluate the instance-level weighting scheme of NESTEDMAML. We follow the classification experiments in (Finn, Abbeel,

$\mathcal{D}_{out}$	5-way 3-shot SVHN			FashionMNIST		
OOD Ratio	30%	60%	90%	30%	60%	90%
MAML-OOD-RM(Skyline)	57.73±0.76	55.29±0.78	54.38±0.12	56.78±0.75	55.29±0.78	53.43±0.51
MAML	55.41±0.75	53.93±0.76	44.10±0.68	54.65±0.77	54.52±0.76	41.52±0.74
MMAML	51.04±0.87	50.28±0.97	41.56±0.96	50.32±0.93	47.54±1.05	42.09±0.97
B-TAML	53.87±0.18	49.84±0.23	42.00±0.21	51.14±0.23	46.59±0.20	36.69±0.21
L2R	47.13±0.13	40.69±0.62	47.26±0.72	33.14±0.60	44.03±0.70	33.06±0.60
Transductive Fine-tuning	55.36±0.73	54.08±0.47	45.21±0.54	55.34±0.45	51.12±0.65	47.42±0.82
NESTEDMAML-FO (ours)	54.76±1.19	45.86±1.19	43.55±1.20	**57.00**±1.20	55.18±1.16	48.52±1.21
NESTEDMAML (ours)	**57.12**±0.81	**55.66**±0.78	**52.16**±0.76	56.66±0.78	**56.04**±0.79	**49.71**±0.78

$\mathcal{D}_{out}$	5-way 5-shot SVHN			FashionMNIST		
OOD Ratio	30%	60%	90%	30%	60%	90%
MAML-OOD-RM(Skyline)	61.89±0.69	61.31±0.75	57.79±0.69	59.83±0.76	61.31±0.75	59.61±0.75
MAML	58.90±0.71	58.66±0.75	49.94±0.69	59.06±0.68	59.25±0.73	49.84±0.69
MMAML	52.45±1.00	52.17±1.05	46.51±1.09	51.46±0.91	54.13±0.93	50.27±1.00
B-TAML	58.34±0.20	56.07±0.21	49.84±0.20	55.19±0.20	52.10±0.19	40.02±0.19
L2R	47.11±0.51	48.01±0.70	51.53±0.71	46.03±0.30	49.15±0.68	55.03±0.46
Transductive Fine-tuning	59.16±0.76	57.84±0.58	53.64±0.42	56.54±0.87	56.23±0.70	54.28±0.32
NESTEDMAML-FO (ours)	57.96±0.94	53.66±0.95	47.58±0.96	**60.59**±0.99	**60.55**±0.95	49.23±0.98
NESTEDMAML (ours)	**60.76**±0.70	**60.53**±0.71	**57.88**±0.70	60.41±0.72	**60.54**±0.72	**57.95**±0.71

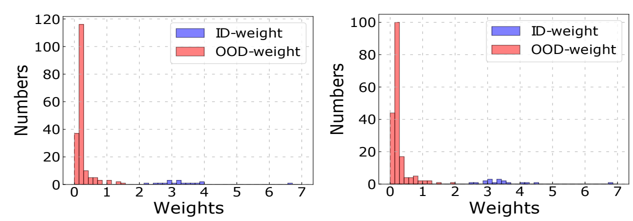
Table 2: Few-shot classification accuracies for the OOD experiment on various evaluation setups. *mini*-Imagenet is used as an in-distribution dataset ($\mathcal{D}_{in}$) for all experiments.

and Levine 2017) to do few-shot learning to evaluate both the task-level and the instance-level weighting schemes. In addition, a synthetic regression experiment is conducted for the task-level weighting scheme as well. Due to the space limitation, we list synthetic regression experiments and detailed experimental settings in Appendix D. We performed all the experiments using PyTorch, and the code is available at https://github.com/Hugo101/NestedMAML.

Datasets. We use ***mini*-ImageNet** (Ravi and Larochelle 2016), **SVHN** (Netzer et al. 2011), **FashionMNIST** (Xiao, Rasul, and Vollgraf 2017) datasets in our experiments. For the task-level weighting scheme, ***mini*-ImageNet** is considered as the ID tasks source ($\mathcal{D}_{in}$). Both the **SVHN** and the **FashionMNIST** datasets are used as OOD tasks source ($\mathcal{D}_{out}$) for ***mini*-ImageNet**. For instance-level weighting, ***mini*-ImageNet** is considered with corrupted labels. Additional details about datasets are given in Appendix D.2.

Task-level Weighting for OOD Tasks

Settings. We implement image classification experiments in 5-way, 3-shot (5-shot) settings. And we use a model with similar backbone architecture given in (Vinyals et al. 2016; Finn, Abbeel, and Levine 2017) for all baselines. We consider a total of 20,000 training tasks containing both ID and OOD tasks where the split of ID and OOD tasks is determined by OOD ratio(0.3, 0.6, and 0.9 in this setting). At each iteration, ID tasks and OOD tasks will be sampled according to the OOD ratio. We sample the ID tasks (meta-training, meta-validation, and meta-test) from the ***mini*-ImageNet** dataset and sample OOD tasks from the **SVHN** or the **FashionMNIST** dataset. We process all images to be of size $84\times84\times3$. As mentioned before, in the task-level weighting, all the data instances in a task share the same weight, reducing the weight hyper-parameters count. To further reduce them, we use the



(a) 5-way 3-shot (b) 5-way 5-shot

Figure 3: Task weight distribution under 90% ratio (SVHN).

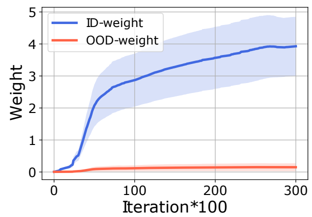
K-means clustering method to cluster the tasks and assign a single weight value to all the same cluster tasks.

Baselines. In addition to **MAML**, we have **MAML-OOD-RM** which basically removes the OOD tasks during meta-training and hence is a skyline to our model. **MMAML** (Vuorio et al. 2019) leverages the strengths of meta-learners by identifying the mode of the task distribution and modulating the meta-learned prior in the parameter space. **B-TAML** (Lee et al. 2020) uses relocated initial parameters for new arriving tasks to handle OOD tasks. We adapted **L2R** (Ren et al. 2018) to assign weights for different tasks and optimize these weights through stochastic gradient descent. We consider **Transductive Fine-tuning** (Dhillon et al. 2019) as a baseline where we finetune the parameters of the model that is obtained by adding a new classifier on top of a pre-trained deep network, which is pre-trained on support and query sets of the meta-training set, using the meta-test set's support and unlabeled query set.

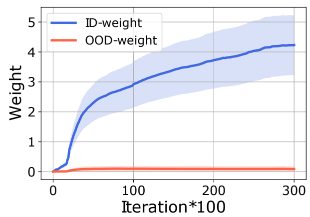
Results in Table 2 show that NESTEDMAML significantly outperforms all baselines and achieves performance competitive to the skyline method (MAML-OOD-RM) in the experiment of SVHN as OOD. For FashionMNIST OOD, NESTEDMAML still outperforms all baseline techniques

Noise Ratio	5-way 3-shot 20%	30%	50%	5-way 5-shot 20%	30%	50%
MAML-Noise-RM	60.2±0.02	59.35±0.01	58.21±0.71	61.2±0.21	60.3±0.32	59.1±0.68
MAML	54.8±0.64	53.9±1.10	51.8±0.12	59.2±0.28	57.6±0.36	53.5±0.48
NESTEDMAML (ours)	**55.24**±0.72	**54.7**±1.20	**53.68**±0.21	**59.6**±0.54	**58.16**±0.87	**55.61**±1.32

Table 3: Test accuracies on *mini*-Imagenet with 20%, 30%, and 50% flipped noisy labels during the meta-training phase.



(a) 5-way 3-shot



(b) 5-way 5-shot

Figure 4: Weights trend as the iterations progress for 30% SVHN OOD experiment

for 60% and 90% ratio. For 30% ratio, the first-order approximation, NESTEDMAML-FO, has the best accuracy, and NESTEDMAML's accuracy is also comparable. Besides, the variance of NESTEDMAML is smaller than NESTEDMAML-FO, which means NESTEDMAML is more stable than NESTEDMAML-FO and NESTEDMAML still has the best performance overall. From the perspective of training time, we observed that NESTEDMAML takes $1.7\times$ and NESTEDMAML-FO takes $1.4\times$ the time taken by MAML for training. Figure 3 shows weight distribution for OOD and ID tasks under 90% ratio when SVHN is viewed as the OOD dataset for 5-way 3-shot (5-shot) settings after the meta-training phase. Both settings show that OOD tasks have much smaller weights than ID tasks: the weights belonging to OOD tasks approximately range from 0 to 1; however, the assigned weights for ID tasks are from 2 to 5, sometimes up to 7.

To showcase the weights adaptation process during the training phase, we plot the weights trend as the iterations progress under the 30% OOD ratio (SVHN) in Figure 4. The Blue (Red) curve denotes the mean weights for ID (OOD) tasks. The shade reflects the variance of weights. Results show that the mean weight assigned to ID tasks would increase as the iterations progress, whereas the weights assigned to OOD tasks remain close to zero, which validates the effectiveness of the NESTEDMAML.

Instance-level Weighting For Noisy Labels

Similar to OOD experiments, we implement 5-way 3-shot (5-shot) experiments to evaluate the instance-level weighting scheme. We conduct experiments on noisy labels generated by randomly corrupting the original labels in ***mini*-ImageNet**. Specifically, different percentages (20%,30%, 50%) of training samples are selected randomly to flip their labels to simulate the noisy corrupted samples. Intuitively, a deep model robust to noise tries to ignore the data with noisy labels. Note that data containing noisy labels only exist in the meta-training stage. Hyper-parameters are shown in Appendix D.2.

Baselines. We compare our NESTEDMAML with the following baselines: (1) **MAML-Noise-RM** serves as a skyline. It is simply modified from MAML, and we manually fix zero weights to instances with noisy labels. (2) **MAML**.

Results in Table 3 show that NESTEDMAML performs better than MAML with high accuracies. Furthermore, to circumvent overfitting and reduce computational complexity due to the weight matrix's high dimension, we group instance weights with 200 clusters by K-means, where instances in each cluster share the same weight initialized at 0.005.

Sensitivity Analysis

We show an ablation study to determine how the number of hyper-parameters and meta-validation sets' size can affect the NESTEDMAML algorithm's performance in Appendix D.2.

5-Way 1-Shot Experiment

We show 5-way 1-shot experiments for two OOD datasets for 30%, 60% in Table 7 in Appendix(using the same hyperparameters). The results show that NESTEDMAML still outperforms MAML by around 2%.

Conclusion

We propose a novel robust meta-learning algorithm for reweighting tasks/instances of corrupted data in the meta-training phase. Our method is model-agnostic, can be directly applied to any deep learning architecture in an end-to-end manner. To the best of our knowledge, NESTEDMAML is the first algorithm to solve a *nested bi-level* optimization problem in an online manner with a convergence result. Finally, empirical evaluation results in OOD task and noisy label scenarios show that NESTEDMAML outperforms state-of-the-art meta-learning methods by efficiently mitigating the effects of unwanted instances or tasks.

Acknowledgments

We thank the AAAI area chairs and anonymous reviewers for offering their constructive comments on this paper. RI and KK were supported by the National Science Foundation(NSF) under Grant Number 2106937, a startup grant from UT Dallas, as well as Google and Adobe awards. FC and CL were funded by the National Science Foundation (NSF) under grants 1815696, 1750911, and 2107449. The opinions, findings, conclusions, and recommendations expressed in this material are those of the author(s). They do not necessarily reflect the views of the National Science Foundation, Google, and Adobe.

References

Balcan, M.-F.; Khodak, M.; and Talwalkar, A. 2019. Provable guarantees for gradient-based meta-learning. In *International Conference on Machine Learning*, 424–433. PMLR.

Behl, H. S.; Baydin, A. G.; and Torr, P. H. 2019. Alpha MAML: Adaptive Model-Agnostic Meta-Learning. *arXiv preprint arXiv:1905.07435*.

Cai, D.; Sheth, R.; Mackey, L.; and Fusi, N. 2020. Weighted Meta-learning. *arXiv preprint arXiv:2003.09465*.

Chen, W.-Y.; Liu, Y.-C.; Kira, Z.; Wang, Y.-C. F.; and Huang, J.-B. 2019. A Closer Look at Few-shot Classification. *International Conference on Learning Representations (ICLR)*.

Dhillon, G. S.; Chaudhari, P.; Ravichandran, A.; and Soatto, S. 2019. A baseline for few-shot image classification. *arXiv preprint arXiv:1909.02729*.

Fallah, A.; Mokhtari, A.; and Ozdaglar, A. 2020. On the convergence theory of gradient-based model-agnostic meta-learning algorithms. In *International Conference on Artificial Intelligence and Statistics*, 1082–1092. PMLR.

Finn, C.; Abbeel, P.; and Levine, S. 2017. Model-agnostic meta-learning for fast adaptation of deep networks. In *Proceedings of the 34th International Conference on Machine Learning-Volume 70*, 1126–1135. JMLR. org.

Finn, C.; Rajeswaran, A.; Kakade, S.; and Levine, S. 2019. Online Meta-Learning. In *International Conference on Machine Learning*, 1920–1930.

Finn, C.; Xu, K.; and Levine, S. 2018. Probabilistic model-agnostic meta-learning. In *Advances in Neural Information Processing Systems*, 9516–9527.

Goldblum, M.; Fowl, L.; and Goldstein, T. 2020. Adversarially Robust Few-Shot Learning: A Meta-Learning Approach. *Advances in Neural Information Processing Systems*, 33.

Jalal, A.; Ilyas, A.; Daskalakis, C.; and Dimakis, A. G. 2017. The robust manifold defense: Adversarial training using generative models. *arXiv preprint arXiv:1712.09196*.

Jeong, T.; and Kim, H. 2020. OOD-MAML: Meta-learning for few-shot out-of-distribution detection and classification. *Advances in Neural Information Processing Systems*, 33.

Killamsetty, K.; Sivasubramanian, D.; Ramakrishnan, G.; and Iyer, R. 2020. GLISTER: Generalization based Data Subset Selection for Efficient and Robust Learning. *arXiv preprint arXiv:2012.10630*.

Lee, H. B.; Lee, H.; Na, D.; Kim, S.; Park, M.; Yang, E.; and Hwang, S. J. 2020. Learning to Balance: Bayesian Meta-Learning for Imbalanced and Out-of-distribution Tasks. In *ICLR*.

Lu, J.; Jin, S.; Liang, J.; and Zhang, C. 2020. Robust Few-Shot Learning for User-Provided Data. *IEEE Transactions on Neural Networks and Learning Systems*.

Luo, Y.; Boix, X.; Roig, G.; Poggio, T.; and Zhao, Q. 2015. Foveation-based mechanisms alleviate adversarial examples. *arXiv preprint arXiv:1511.06292*.

Naik, D. K.; and Mammone, R. 1992. Meta-neural networks that learn by learning. *[Proceedings 1992] IJCNN International Joint Conference on Neural Networks*, 1: 437–442 vol.1.

Netzer, Y.; Wang, T.; Coates, A.; Bissacco, A.; Wu, B.; and Ng, A. Y. 2011. Reading digits in natural images with unsupervised feature learning. *NIPS Workshop on Deep Learning and Unsupervised Feature Learning*.

Nichol, A.; Achiam, J.; and Schulman, J. 2018. On first-order meta-learning algorithms. *arXiv preprint arXiv:1803.02999*.

Raghu, A.; Raghu, M.; Bengio, S.; and Vinyals, O. 2019. Rapid learning or feature reuse? towards understanding the effectiveness of maml. *arXiv preprint arXiv:1909.09157*.

Rajeswaran, A.; Finn, C.; Kakade, S. M.; and Levine, S. 2019. Meta-learning with implicit gradients. In *Advances in Neural Information Processing Systems*, 113–124.

Ravi, S.; and Larochelle, H. 2016. Optimization as a model for few-shot learning. *ICLR*.

Ren, M.; Zeng, W.; Yang, B.; and Urtasun, R. 2018. Learning to reweight examples for robust deep learning. In *International Conference on Machine Learning*, 4334–4343. PMLR.

Rusu, A. A.; Rao, D.; Sygnowski, J.; Vinyals, O.; Pascanu, R.; Osindero, S.; and Hadsell, R. 2018. Meta-learning with latent embedding optimization. *arXiv preprint arXiv:1807.05960*.

Santoro, A.; Bartunov, S.; Botvinick, M.; Wierstra, D.; and Lillicrap, T. 2016. Meta-learning with memory-augmented neural networks. In *International conference on machine learning*, 1842–1850.

Schmidhuber, J. 1987. *Evolutionary Principles in Self-Referential Learning. On Learning now to Learn: The Meta-Meta-Meta...-Hook*. Diploma thesis, Technische Universitat Munchen, Germany.

Schneider, S.; Rusak, E.; Eck, L.; Bringmann, O.; Brendel, W.; and Bethge, M. 2020. Improving robustness against common corruptions by covariate shift adaptation. *Advances in Neural Information Processing Systems*, 33.

Shu, J.; Xie, Q.; Yi, L.; Zhao, Q.; Zhou, S.; Xu, Z.; and Meng, D. 2019. Meta-weight-net: Learning an explicit mapping for sample weighting. In *Advances in Neural Information Processing Systems*, 1919–1930.

Triantafillou, E.; Zhu, T.; Dumoulin, V.; Lamblin, P.; Evci, U.; Xu, K.; Goroshin, R.; Gelada, C.; Swersky, K.; Manzagol, P.-A.; et al. 2020. Meta-dataset: A dataset of datasets for learning to learn from few examples. *ICLR*.

Vinyals, O.; Blundell, C.; Lillicrap, T.; Wierstra, D.; et al. 2016. Matching networks for one shot learning. In *Advances in neural information processing systems*, 3630–3638.

Vuorio, R.; Sun, S.-H.; Hu, H.; and Lim, J. J. 2019. Multi-modal Model-Agnostic Meta-Learning via Task-Aware Modulation. In *Advances in Neural Information Processing Systems*, 1–12.

Wang, H.; and Yu, C.-N. 2019. A direct approach to robust deep learning using adversarial networks. *arXiv preprint arXiv:1905.09591*.

Wei, H.; and Kehtarnavaz, N. 2020. Simultaneous utilization of inertial and video sensing for action detection and recognition in continuous action streams. *IEEE Sensors Journal*, 20(11): 6055–6063.

Xiao, H.; Rasul, K.; and Vollgraf, R. 2017. Fashion-mnist: a novel image dataset for benchmarking machine learning algorithms. *arXiv preprint arXiv:1708.07747*.

Yao, H.; Wu, X.; Tao, Z.; Li, Y.; Ding, B.; Li, R.; and Li, Z. 2020. Automated relational meta-learning. *ICLR*.

Yin, C.; Tang, J.; Xu, Z.; and Wang, Y. 2018. Adversarial meta-learning. *arXiv preprint arXiv:1806.03316*.

Zhao, C.; Chen, F.; and Thuraisingham, B. 2021. Fairness-Aware Online Meta-Learning. In *Proceedings of the 27th ACM SIGKDD Conference on Knowledge Discovery; Data Mining*. ISBN 9781450383325.

Zhao, C.; Chen, F.; Wang, Z.; and Khan, L. 2020a. A Primal-Dual Subgradient Approach for Fair Meta Learning. *Proceedings of the 20th IEEE International Conference on Data Mining (ICDM)*.

Zhao, C.; Li, C.; Li, J.; and Chen, F. 2020b. Fair meta-learning for few-shot classification. In *2020 IEEE International Conference on Knowledge Graph (ICKG)*, 275–282. IEEE.

Zhou, A.; Knowles, T.; and Finn, C. 2021. Meta-learning Symmetries by Reparameterization. In *International Conference on Learning Representations*.