Sign-MAML: Efficient Model-Agnostic Meta-Learning by SignSGD

Chen Fan
College of Information and Computer Sciences,
University of Massachusetts Amherst

Parikshit Ram
IBM Research

Sijia Liu
Computer Science and Engineering, Michigan State University
MIT-IBM Watson AI Lab

Abstract

We propose a new computationally-efficient first-order algorithm for Model-Agnostic Meta-Learning (MAML). The key enabling technique is to interpret MAML as a bilevel optimization (BLO) problem and leverage the sign-based SGD (signSGD) as a lower-level optimizer of BLO. We show that MAML, through the lens of signSGD-oriented BLO, naturally yields an alternating optimization scheme that just requires first-order gradients of a learned meta-model. We term the resulting MAML algorithm *Sign-MAML*. Compared to the conventional first-order MAML (FO-MAML) algorithm, Sign-MAML is theoretically-grounded as it does not impose any assumption on the absence of second-order derivatives during meta training. In practice, we show that Sign-MAML outperforms FO-MAML in various few-shot image classification tasks, and compared to MAML, it achieves a much more graceful tradeoff between classification accuracy and computation efficiency.

1 Introduction

Humans can learn new tasks quickly based on prior knowledge or experience with similar tasks. A meta-learning algorithm resembles this in a way such that given previous exposure to relevant tasks, new tasks can be learned with a small amount of data. To do this, it involves a meta(or upper)-learner whose job is to update parameters of a base(or lower)-learner which aims to solve a specific task (e.g. image classification) at hand. This 'learning to learn' hierarchical structure can be viewed as solving a bilevel optimization (BLO) problem, in which the solution to the lower-level problem provides useful feedback for updating the solution of an upper-level problem [1, 2]. Recent works have studied the optimization-based meta-learning approach targeting on different parameters associated with the base learner, such as learning a good weight initialization [3–5], and updating neural network architectures [6–8].

Within the optimization-based meta-learning family, Model-Agnostic Meta-Learning (MAML) is a popular method that has been widely applied to solving computer vision and natural language processing tasks [9–11]. In-depth empirical and theoretical understanding of MAML has also been provided in [4, 12–14]. Through the lens of BLO, MAML is composed of an upper-level optimization step (which updates weight initialization of a model), and a sequence of lower-level steps (which adapt this initialization to different specific tasks). Despite the effectiveness of MAML, it is difficult to scale to large models and datasets due to the need of second-order derivatives during model training [4, 15]. A first-order variant (FO-MAML) solves this problem by ignoring [3] or estimating [13] second-order derivatives in practice at the cost of introducing meta-gradient estimation error.

5th Workshop on Meta-Learning at NeurIPS 2021, Sydney, Australia.

Contributions In this work, we aim to design a computationally-efficient and theoretically-grounded MAML algorithm that only relies on first-order derivatives in its implementation. To this end, we propose Sign-MAML by integrating MAML with signSGD [16] and show its advantages over FO-MAML and MAML. Our contributions are summarized below:

- (Formulation-wise) In §3, we revisit MAML through the lens of BLO and identify a tight connection between its computation efficiency and the choice of a lower-level optimizer.
- (Methodology-wise) In §4, we leverage signSGD to unroll the lower-level problem of MAML and theoretically show that this naturally leads to a first-order alternating optimization method, *Sign-MAML*, whose computation is exactly as efficient as FO-MAML.
- (Application-wise) In §5, we conduct extensive experiments to demonstrate the advantage of Sign-MAML in computation efficiency and accuracy. In particular, we show that Sign-MAML has computational costs similar to those of FO-MAML while providing significantly improved accuracy for few-shot tasks.

2 Related Work

Meta-learning A surge of recent works have been devoted to developing theory and algorithms of MAML [2, 5, 13, 14, 17]. For example, the 'Almost No Inner Loop' (ANIL) algorithm was proposed in [5], which dissects the meta-learning into two phases: training the initialization of a meta-model, and partially fine-tuning the classification head of the meta-model. Compared to the conventional MAML algorithm, ANIL yields a reduced computation cost due to the use of partial fine-tuning instead of the end-to-end full fine-tuning. However, ANIL still needs second-order derivatives during meta training. To overcome such a computation bottleneck, the work [13, 17] proposed to use the finite difference of function values or first-order gradients to estimate the high-order derivatives involved in MAML. However, the resulting gradient/Hessian estimation may not be unbiased and could lead to an unexpected large estimation variance [18]. Another first-order method proposed in [4] simplifies meta gradient computation by using the difference between initial and adapted weights. Here, we focus on the design of lower-level optimizer to speed up computation. Besides optimization-based meta-learning, other algorithms such as metric-based and model-based meta-learning have also been developed [2, 19–23]. In this work, we focus on the optimization-based meta-learning.

Bilevel optimization Bilevel optimization is applied to solve problems that exhibit two-level hierarchical structure in which the solution to the lower-level problem is an input to the upper-level problem. Solvers for BLO problems can be either deterministic or stochastic. Under deterministic BLO appraoch, two commonly used methods are approximate implicit differentiation (AID) based and iterative differentiation (ITD) based. For both methods, the lower-level problem is solved by gradient descent (GD). For the upper-level problem, AID-based methods obtain meta-gradients through implicit gradients [24] whereas ITD-based methods rely on backpropagation [25, 26]. In recent years, stochastic approaches have gained a lot of attentions due to its fast convergence and scalability. Ghadimi and Wang [27] proposed a method to obtain lower and upper-level gradients through stochastic approximations. Hong et al. [28] developed a method that solves lower and upper-level problems simultaneously with lower and upper-level step sizes at two different scales.

Sign-based optimization methods signSGD [16] utilizes the sign of gradients as the descent direction for FO non-convex optimization and demonstrates a convergence rate comparable to that of stochastic gradient descent (SGD). Liu et al. [29] proposed zeroth-order signSGD (ZO-signSGD) for solving optimization problems where first-order derivatives are difficult or infeasible to obtain, demonstrating lower estimation variance when compared to conventional ZO-SGD schemes [18, 30]. In adversarial machine learning, fast gradient sign method (FGSM) has been commonly used for generating prediction-evasion adversarial attacks [31] and for training an adversarially robust deep neural network [32].

3 Problem Statement

In this section, we begin by presenting the problem of MAML and interpreting it through the lens of BLO (bilevel optimization). Next, we illustrate the limitations of existing solutions to MAML and elaborate on our research objective.

BLO setup of MAML Considering P tasks where each task $\tau_i, i \in [P]$ is sampled from a task distribution, MAML seeks to solve the following problem from a bilevel optimization perspective:

$$\begin{aligned} \underset{\mathbf{x}}{\text{minimize}} \quad & L(\mathbf{x}) := \tfrac{1}{P}\textstyle\sum_{i=1}^{P} \ell_i(\mathbf{y}_i^*(\mathbf{x}); \mathcal{D}_i^{\texttt{val}}) \\ \text{subject to} \quad & \mathbf{y}_i^*(\mathbf{x}) \in \arg\min_{\mathbf{y}'} \ell_i(\mathbf{y}'; \mathcal{D}_i^{\texttt{tr}}), \quad \forall i \in [P] = \{1, \ldots, P\}, \end{aligned} \tag{1}$$

where $\mathbf{x}$ is the weight initialization of a model, $\mathbf{y}_i^*(\mathbf{x})$ is the optimal weight after the model is fine-tuned for task i with ℓ_i, $\mathcal{D}_i^{\texttt{tr}}$ and $\mathcal{D}_i^{\texttt{val}}$ as the task-specific loss, training (support) set and validation (query) set respectively.

A generic BLO formulation of the MAML problem (1) is then given by

$$\underset{\mathbf{x}}{\text{minimize}} \ \underbrace{f(\mathbf{x}, \mathbf{y}^*(\mathbf{x}))}_{\text{Upper-level problem}} \ \text{subject to} \underbrace{\mathbf{y}^*(\mathbf{x}) \in \arg\min_{\mathbf{y}} g(\mathbf{x}, \mathbf{y})}_{\text{Lower-level problem}}, \tag{2}$$

where a lower-level solution is used as an input to minimize the upper-level objective f. The MAML problem (1) is a special case of the BLO formulation (2): The upper-level objective function $L(\mathbf{x})$ in (1) is not an exact bi-variate function $f(\mathbf{x}, \mathbf{y})$ as (2); Instead, $L(\mathbf{x})$ relies only on a lower-level solution $\mathbf{y}^*$, which is a function of $\mathbf{x}$.

Second-order derivatives requested in MAML Conventionally, MAML [3] solves the lower-level problem of (1) through a m-step **SGD unrolling**. Let $\ell_i(\mathbf{y}') := \ell_i(\mathbf{y}'; \mathcal{D}_i^{\texttt{val}})$, $\hat{\ell}_i(\mathbf{y}') := \ell_i(\mathbf{y}'; \mathcal{D}_i^{\texttt{tr}})$, and $\mathbf{x}_k$ denote the model initialization at the kth upper-level iteration, the original MAML algorithm is then given by

$$\text{Lower-level: } \underbrace{\mathbf{y}_i^{(0)}(\mathbf{x}_k) = \mathbf{x}_k; \ \ \mathbf{y}_i^{(m)}(\mathbf{x}_k) = \mathbf{y}_i^{(m-1)}(\mathbf{x}_k) - \beta \, \nabla_{\mathbf{y}'} \hat{\ell}_i(\mathbf{y}')\Big|_{\mathbf{y}_i^{(m-1)}(\mathbf{x}_k)}}_{m\text{-step SGD unrolling}} \tag{3}$$

$$\text{Upper-level: } \mathbf{x}_{k+1} = \mathbf{x}_k - \alpha \frac{1}{P} \sum_{i=1}^{P} \nabla_{\mathbf{x}} \ell_i(\mathbf{y}_i^{(m)}(\mathbf{x}_k)), \tag{4}$$

where $\alpha, \beta > 0$ are learning rates of SGD used for upper-level and lower-level optimization, respectively. Substituting the lower-level SGD unrolling into the upper-level SGD step, the overall optimization step to update the optimizee variable $\mathbf{x}$ is given by [3, 4]

$$\mathbf{x}_{k+1} = \mathbf{x}_k - \alpha \underbrace{\frac{1}{P} \sum_{i=1}^{P} \prod_{n=0}^{m-1} \left(\mathbf{I} - \beta \, \nabla^2_{\mathbf{y}'} \hat{\ell}_i(\mathbf{y}')\Big|_{\mathbf{y}'=\mathbf{y}_i^{(n)}(\mathbf{x}_k)} \right) \nabla_{\mathbf{y}'} \ell_i(\mathbf{y}')\big|_{\mathbf{y}'=\mathbf{y}_i^{(m)}(\mathbf{x}_k)}}_{\text{Meta-gradient w.r.t. } \mathbf{x}}, \tag{5}$$

where $\nabla^2_{\mathbf{y}'}$ denotes the second-order derivatives with respect to (w.r.t.) the variable $\mathbf{y}'$. In (5), the computation involving $\nabla^2_{\mathbf{y}'} \hat{\ell}_i(\mathbf{y}')$ is **costly** for large neural networks and datasets, and this cost increases with the number of fine-tuning steps.

Research objective To resolve the difficulty induced by second-order derivatives, FO-MAML **assumes** them to be **0** in the computation [3]. This would introduce an error into the meta-gradient in (5), and may hamper its generalization ability. In fact, a generalization gap as large as 6% is observed for tasks such as 20-way 1-shot on Omniglot dataset between MAML and FO-MAML [15]. Fallah et al. [13] present a Hessian-free MAML which has improved theoretical convergence over FO-MAML but its empirical performance or efficiency has not been studied. From (5), we see the cause of high computation cost is rooted in the coupling between lower and upper-level problems in which backpropagation has to loop through the entire lower-level optimization trajectory. To bypass this, Rajeswaran et al. [15] proposed implicit MAML (iMAML) to directly solve the BLO problem (2) using the implicit gradient method. However, iMAML needs (an approximation of) a matrix inversion operation to calculate an implicit gradient. Among the aforementioned algorithms, FO-MAML is the computationally lightest but yields a poorer optimization accuracy. By contrast, MAML and iMAML have improved generalization ability but higher computational costs. Spurred by above, we ask:

How to develop an assumption-least first-order MAML algorithm that enjoys the dual advantages of low computation cost and high optimization accuracy?

4 Sign-MAML: Advancing MAML by SignSGD

In this section, we first present the method of signSGD unrolling to solve the BLO problem (2). Then we apply the achieved results to the case of MAML to establish our Sign-MAML method. At the end, we highlight the differences between FO-MAML and Sign-MAML.

BLO solver based on signSGD unrolling We propose to unroll the lower-level problem in (2) using signSGD [16]. The last step of a m-step unrolling via signSGD is given by

$$\mathbf{y}^{(m)} = \mathbf{y}^{(m-1)} - \beta \text{sign}(\nabla_{\mathbf{y}} g(\mathbf{x}, \mathbf{y}^{(m-1)})), \tag{6}$$

where $\text{sign}(\cdot)$ denotes element-wise sign operation, $\beta > 0$ is the lower-level learning rate, and $\mathbf{y}^{(0)}$ can be a random starting point. Substituting (6) into problem (2) with $\mathbf{y}^*(\mathbf{x}) = \mathbf{y}^{(m)}$, we have the following variant of the original BLO problem

$$\text{minimize} \quad f(\mathbf{x}, \mathbf{y}^{(m)}(\mathbf{x})), \tag{7}$$

where we explicitly express $\mathbf{y}^{(m)}$ as a function of $\mathbf{x}$. To optimize $\mathbf{x}$, we resort to GD/SGD

$$\mathbf{x}_{k+1} = \mathbf{x}_k - \alpha \frac{df(\mathbf{x}_k, \mathbf{y}^{(m)}(\mathbf{x}_k))}{d\mathbf{x}}, \tag{8}$$

where $\alpha > 0$ is the upper-level learning rate and k is the descent step index. In (8), the key step is to compute the gradient w.r.t. $\mathbf{x}$, namely,

$$\frac{df(\mathbf{x}_k, \mathbf{y}^{(m)}(\mathbf{x}_k))}{d\mathbf{x}} = \frac{\partial f(\mathbf{x}_k, \mathbf{y}^{(m)}(\mathbf{x}_k))}{\partial \mathbf{x}} + \frac{d\mathbf{y}^{(m)}(\mathbf{x}_k)}{d\mathbf{x}}^\top \frac{\partial f(\mathbf{x}_k, \mathbf{y}^{(m)}(\mathbf{x}_k))}{\partial \mathbf{y}}, \tag{9}$$

where $\frac{\partial f(\mathbf{x},\mathbf{y})}{\partial \mathbf{x}} = \nabla_{\mathbf{x}} f$ and $\frac{df(\mathbf{x},\mathbf{y})}{d\mathbf{x}}$ denote the partial and full derivatives of f w.r.t. the variable $\mathbf{x}$ respectively. Based on (6) and the **key fact** that $\frac{d\text{sign}(\mathbf{z})}{d\mathbf{z}}^\top = \mathbf{0}$ (holding almost surely), we have

$$\frac{d\mathbf{y}^{(m)}(\mathbf{x}_k)}{d\mathbf{x}}^\top = \frac{d\mathbf{y}^{(m-1)}(\mathbf{x}_k)}{d\mathbf{x}}^\top = \cdots = \frac{d\mathbf{y}^{(0)}(\mathbf{x}_k)}{d\mathbf{x}}^\top. \tag{10}$$

Substituting (10) into (9), we achieve

$$\frac{df(\mathbf{x}_k, \mathbf{y}^{(m)}(\mathbf{x}_k))}{d\mathbf{x}} = \frac{\partial f(\mathbf{x}_k, \mathbf{y}^{(m)}(\mathbf{x}_k))}{\partial \mathbf{x}} + \frac{d\mathbf{y}^{(0)}(\mathbf{x}_k)}{d\mathbf{x}}^\top \frac{\partial f(\mathbf{x}_k, \mathbf{y}^{(m)}(\mathbf{x}_k))}{\partial \mathbf{y}}. \tag{11}$$

This implies that when signSGD is used to unroll the lower-level problem, we can naturally reach a **first-order alternating optimization** method:

$$\mathbf{y}\text{-step: signSGD unrolling (6)} \tag{12}$$

$$\mathbf{x}\text{-step: } \mathbf{x}_{k+1} = \mathbf{x}_k - \alpha \frac{\partial f(\mathbf{x}_k, \mathbf{y}^{(m)}(\mathbf{x}_k))}{\partial \mathbf{x}} - \alpha \frac{d\mathbf{y}^{(0)}(\mathbf{x}_k)}{d\mathbf{x}}^\top \frac{\partial f(\mathbf{x}_k, \mathbf{y}^{(m)}(\mathbf{x}_k))}{\partial \mathbf{y}}. \tag{13}$$

Sign-MAML: MAML based on signSGD unrolling We now apply (12) and (13) to the case of MAML in which $\mathbf{y}^{(0)}(\mathbf{x}_k) = \mathbf{x}_k$ and the dependence of upper objective of MAML on $\mathbf{x}$ is only through $\mathbf{y}(\mathbf{x})$. They lead to $\frac{d\mathbf{y}^{(0)}(\mathbf{x}_k)}{d\mathbf{x}}^\top = \mathbf{I}$ and $\frac{\partial f(\mathbf{x}_k, \mathbf{y}^{(m)}(\mathbf{x}_k))}{\partial \mathbf{x}} = \mathbf{0}$. These two simplifications render (13) to

$$\mathbf{x}_{k+1} = \mathbf{x}_k - \alpha \frac{1}{P} \sum_{i=1}^{P} \nabla_{\mathbf{y}'} \ell_i(\mathbf{y}')|_{\mathbf{y}' = \mathbf{y}_i^{(m)}(\mathbf{x}_k)}. \tag{14}$$

At the first glance, the upper-level MAML update with signSGD unrolling is the same as FO-MAML. However, the **key difference** lies in the **choice of lower-level optimizer**: signSGD unrolling **naturally** leads to (14). By contrast, FO-MAML requires making the **assumption** of $\nabla^2_{\mathbf{y}'} \hat{\ell}_i(\mathbf{y}') = \mathbf{0}$ in (5). Hence, we regard using signSGD unrolling as the 'authentic' first-order method, and we name our algorithm as Sign-MAML shown in Algorithm 1. We also remark that in addition to signSGD, the

gradient sign-based momentum method [16] is another possible alternative to generate the first-order MAML approach via gradient unrolling.

Algorithm 1 Sign-MAML

for $k = 1, 2, \ldots$ **do**
  Sample P tasks from a task distribution
  for $i = 1, 2, \ldots, P$ **do**
    Initialize $\mathbf{y}_i^{(0)}(\mathbf{x}_k) = \mathbf{x}_k$
    Obtain $\mathbf{y}_i^{(m)}$ by signSGD unrolling (6)
  end for
  Compute $\mathbf{x}_{k+1}$ using (14);
end for

5 Experiments

The central questions that we aim to address with our experiments are: ① *Can Sign-MAML perform better than FO-MAML without increasing computation cost?* ② *Can Sign-MAML perform comparably to MAML but with less computation time?* To this end, we measure the *test accuracy* and *train time per upper-iteration* of Sign-MAML, together with baselines (FO-MAML and MAML) in N-way and K-shot image classification tasks.

5.1 Experiment setup

Datesets We conduct experiments on Fewshot-CIFAR100 (FS-CIFAR100) and MiniImageNet datasets [22, 33]. The FS-CIFAR100 dataset has 600 images of size 32×32 in each of the 100 classes from CIFAR100 [34]. We partition the 100 classes into 60 classes , 20 classes and 20 classes for train, validation and test respectively following Oreshkin et al. [33]. The MiniImageNet dataset has 600 images of size 84×84 in each of the 100 classes. We partition the 100 classes into 64 classes, 16 classes and 20 classes for train, validation and test respectively following Ravi and Larochelle [35].

Architectures For MiniImageNet, we use a neural network consisting of 4 convolutional layers with 32 filters in each layer used in Ravi and Larochelle [35]. For FS-CIFAR100, we also use the 4-layer convolutional neural newtork but with 64 filters in each layer. For both neural networks, each convolution operation is followed by batch normalization, ReLU activation and 2×2 max pooling.

Implementation details We use $\alpha = 0.001$ as the upper-level learning rate, $m = 1$ fine-tuning step for training (unless otherwise specified), $m = 10$ fine-tuning steps for testing, and $P = 4$ as the batch size of tasks across all experiments. To setup the lower-level learning rate β, since different optimizers (e.g., signSGD vs. SGD) are used, we perform a grid search on β and pick the one with the best validation performance (see Appendix A for details). We utilized the existing implementation of FO-MAML and MAML in the `learn2learn` Python package [36] and adapt it to implement Sign-MAML.[1]

5.2 Results and Discussions

In what follows, we first show the results of Sign-MAML, FO-MAML and MAML on FS-CIFAR100 for different N-way and K-shot classification tasks. We then provide a detailed comparison between Sign-MAML and FO-MAML, which fall into the first-order optimization category, given various choices of N and K on MiniImageNet. Furthermore, we show the effectiveness of Sign-MAML when different fine-tuning steps are used.

FS-CIFAR100 results Table 1 presents the performance of Sign-MAML versus MAML and FO-MAML on FS-CIFAR100 for 5-way 1-shot classification, 5-way 5-shot classification, 10-way 1-shot classification and 10-way 5-shot classification. Compared to MAML, Sign-MAML performs slightly

[1] Our codes are available at `https://github.com/chenfan95/Sign-MAML`

Table 1: FS-CIFAR100 classification results, which include accuracy (upper-level numbers) and computation time per meta-iteration in seconds (lower-level numbers). For accuracy, the $\pm$ shows 95% confidence intervals over 1000 test-time tasks. For computation time, the $\pm$ shows standard deviation over 1000 meta iterations.

Scenario	MAML	FO-MAML	Sign-MAML
5-way 1-shot	$35.8 \pm 1.4\ \%$ 0.058 ± 0.003	$32.7 \pm 1.3\ \%$ 0.032 ± 0.003	$37.5 \pm 1.4\ \mathbf{\%}$ 0.032 ± 0.003
5-way 5-shot	$48.8 \pm 0.7\ \%$ 0.073 ± 0.008	$45.8 \pm 0.8\ \%$ 0.048 ± 0.006	$49.5 \pm 0.7\ \mathbf{\%}$ 0.049 ± 0.006
10-way 1-shot	$20.9 \pm 0.8\ \%$ 0.064 ± 0.004	$21.4 \pm 0.8\ \%$ 0.039 ± 0.003	$22.5 \pm 0.8\ \mathbf{\%}$ 0.039 ± 0.003
10-way 5-shot	$29.9 \pm 0.4\ \%$ 0.106 ± 0.016	$30.9 \pm 0.4\ \mathbf{\%}$ 0.067 ± 0.015	$30.5 \pm 0.5\ \%$ 0.067 ± 0.016

better for all tasks and takes only half computation time per iteration. Compared to FO-MAML, Sign-MAML achieves a remarkable increase of 4.8% and 3.7% in accuracy for 5-way 1-shot classification and 5-way 5-shot classification, respectively. Moreover, it has very similar computation cost as FO-MAML for all tasks. Overall, Sign-MAML is a competitive method in both performance and computation efficiency when compared to FO-MAML and MAML.

Sign-MAML vs. FO-MAML In Figure 1, we present the classification accuracy of Sign-MAML and FO-MAML in a variety of few-shot learning setup, with $N \in \{2, 5, 7, 10\}$ ways and $K \in \{1, 2, 3, 4, 5\}$ shots on MiniImageNet. We compare algorithms in the computation-lightest regime using 1 gradient unrolling step. As we can see, if the tasks become more challenging, namely, with higher N and lower K, then Sign-MAML performs much better than FO-MAML. For example, Sign-MAML achieves an accuracy that is 7.5% higher than FO-MAML for 10-way 1-shot classification. The performance gap becomes larger as N increases or K decreases. Moreover, for the cases where FO-MAML outperforms Sign-MAML, the performance gaps (1 - 3%) are smaller than the cases where Sign-MAML outperforms FO-MAML (1 - 8 %). The above results suggest that Sign-MAML can be a better approach when challenging tasks are present.

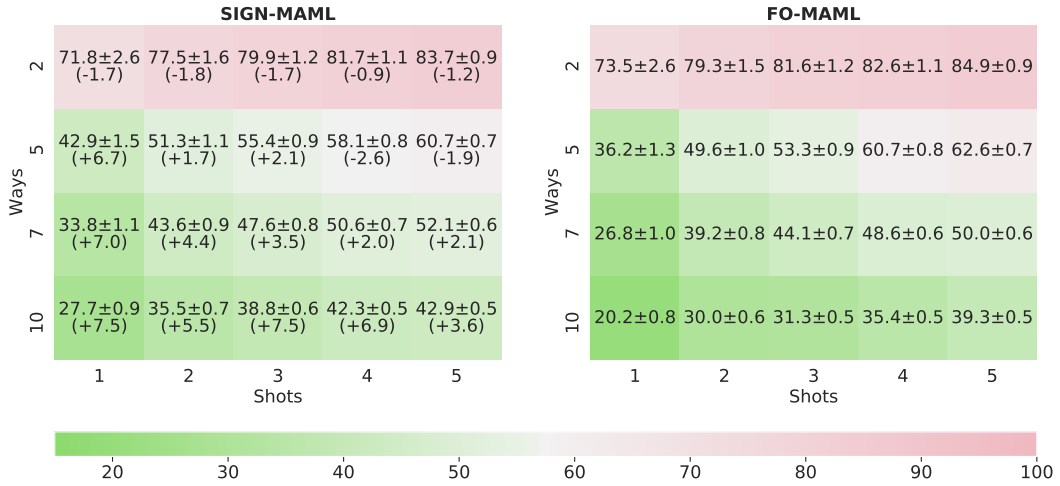
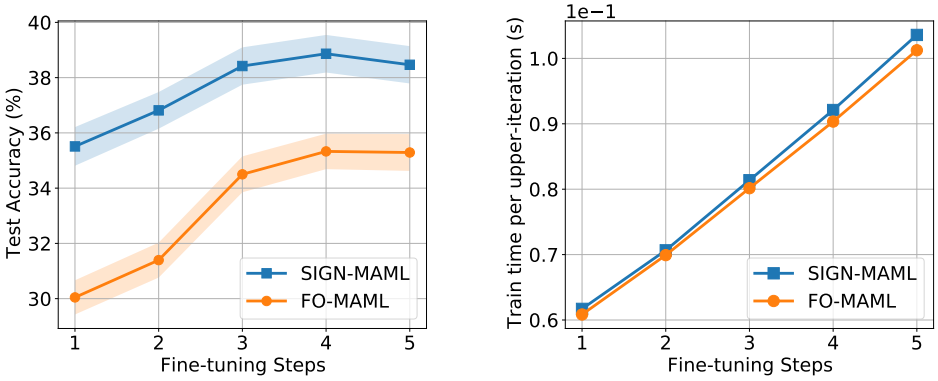


Figure 1: MiniImageNet classification results of Sign-MAML and FO-MAML for different ways and shots. Numbers in each cell are accuracy with its 95 % confidence interval over test tasks. Numbers inside bracket represent the performance improvement (+) or degradation ($-$) of Sign-MAML over FO-MAML. A green or red region indicates the scenario in which Sign-MAML is better or worsen than FO-MAML in accuracy.

Meta-learning vs. fine-tuning steps In Figure 2, we present the classification accuracy as well as the computation cost versus the number of fine-tuning steps. Here we focus on the case of

10-way 2-shot classification on MiniImageNet. It can be seen from Figure 2 (a) that Sign-MAML outperforms FO-MAML at each setup of the fine-tuning step, and the test accuracy increases rapidly at the beginning and saturates towards the end. In addition, Figure 2 (b) shows that the accuracy improvement of Sign-MAML over FO-MAML is *not* at the cost of computation complexity. Clearly, Sign-MAML and FO-MAML take the very similar computation cost, which increases linearly with the number of fine-tuning steps (see Appendix C for results of 5-way 2-shot classification and 7-way 2-shot classification on MiniImageNet).



(a) Accuracy 10-way 2-shot (b) Time 10-way 2-shot

Figure 2: 10-way 2-shot MiniImageNet classification against the choice of the number of fine-tuning steps: (a) classification accuracy and (b) computation time, with the same format as Table 1.

Take-away: Based on the aforementioned results, we find that ① *Sign-MAML typically performs better than FO-MAML for challenging tasks without losing computation efficiency.* ② *Sign-MAML can match or exceed the performance of MAML with less computation time.*

6 Conclusion

In this paper, we show that signSGD can be used as an efficient gradient unrolling scheme to advance MAML (model-agnostic meta-learning). Specifically, the study of MAML through the lens of BLO (bilevel optimization) enables us to customize a 'lower-level' optimizer to 'fine-tune' meta model over task-specific losses. We theoretically show that if signSGD is used as the lower-level optimizer, then MAML can be equivalently transformed into the *first-order* alternating optimization method, termed Sign-MAML. Empirically, we also demonstrate that compared to the conventional MAML and FO-MAML approaches, Sign-MAML places a more graceful tradeoff between accuracy and computation cost. Particular, in a series of challenging few-shot image classification tasks (which involve more classes and less data samples), Sign-MAML yields a consistent improvement over baselines.

References

[1] Joaquin Vanschoren. Meta-learning: A survey. *arXiv preprint arXiv:1810.03548*, 2018.

[2] Timothy Hospedales, Antreas Antoniou, Paul Micaelli, and Amos Storkey. Meta-learning in neural networks: A survey. *arXiv preprint arXiv:2004.05439*, 2020.

[3] Chelsea Finn, Pieter Abbeel, and Sergey Levine. Model-agnostic meta-learning for fast adaptation of deep networks. In *International Conference on Machine Learning*, pages 1126–1135. PMLR, 2017.

[4] Alex Nichol, Joshua Achiam, and John Schulman. On first-order meta-learning algorithms, 2018.

[5] Aniruddh Raghu, Maithra Raghu, Samy Bengio, and Oriol Vinyals. Rapid learning or feature reuse? towards understanding the effectiveness of maml. *arXiv preprint arXiv:1909.09157*, 2019.

[6] Barret Zoph and Quoc V Le. Neural architecture search with reinforcement learning. *arXiv preprint arXiv:1611.01578*, 2016.

[7] Thomas Elsken, Benedikt Staffler, Jan Hendrik Metzen, and Frank Hutter. Meta-learning of neural architectures for few-shot learning. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 12365–12375, 2020.

[8] Dongze Lian, Yin Zheng, Yintao Xu, Yanxiong Lu, Leyu Lin, Peilin Zhao, Junzhou Huang, and Shenghua Gao. Towards fast adaptation of neural architectures with meta learning. In *International Conference on Learning Representations*, 2019.

[9] Luisa Zintgraf, Kyriacos Shiarli, Vitaly Kurin, Katja Hofmann, and Shimon Whiteson. Fast context adaptation via meta-learning. In *International Conference on Machine Learning*, pages 7693–7702. PMLR, 2019.

[10] Zi-Yi Dou, Keyi Yu, and Antonios Anastasopoulos. Investigating meta-learning algorithms for low-resource natural language understanding tasks. *arXiv preprint arXiv:1908.10423*, 2019.

[11] Zequn Liu, Ruiyi Zhang, Yiping Song, and Ming Zhang. When does maml work the best? an empirical study on model-agnostic meta-learning in nlp applications. *arXiv preprint arXiv:2005.11700*, 2020.

[12] Antreas Antoniou, Harrison Edwards, and Amos Storkey. How to train your maml. *arXiv preprint arXiv:1810.09502*, 2018.

[13] Alireza Fallah, Aryan Mokhtari, and Asuman Ozdaglar. On the convergence theory of gradient-based model-agnostic meta-learning algorithms. In *International Conference on Artificial Intelligence and Statistics*, pages 1082–1092. PMLR, 2020.

[14] Kaiyi Ji, Jason D Lee, Yingbin Liang, and H Vincent Poor. Convergence of meta-learning with task-specific adaptation over partial parameters. *arXiv preprint arXiv:2006.09486*, 2020.

[15] Aravind Rajeswaran, Chelsea Finn, Sham Kakade, and Sergey Levine. Meta-learning with implicit gradients. 2019.

[16] Jeremy Bernstein, Yu-Xiang Wang, Kamyar Azizzadenesheli, and Animashree Anandkumar. signsgd: Compressed optimisation for non-convex problems. In *International Conference on Machine Learning*, pages 560–569. PMLR, 2018.

[17] Xingyou Song, Wenbo Gao, Yuxiang Yang, Krzysztof Choromanski, Aldo Pacchiano, and Yunhao Tang. Es-maml: Simple hessian-free meta learning. *arXiv preprint arXiv:1910.01215*, 2019.

[18] Sijia Liu, Pin-Yu Chen, Bhavya Kailkhura, Gaoyuan Zhang, Alfred O Hero III, and Pramod K Varshney. A primer on zeroth-order optimization in signal processing and machine learning: Principals, recent advances, and applications. *IEEE Signal Processing Magazine*, 37(5):43–54, 2020.

[19] Gregory Koch, Richard Zemel, Ruslan Salakhutdinov, et al. Siamese neural networks for one-shot image recognition. In *ICML deep learning workshop*, volume 2. Lille, 2015.

[20] Jake Snell, Kevin Swersky, and Richard S Zemel. Prototypical networks for few-shot learning. *arXiv preprint arXiv:1703.05175*, 2017.

[21] Flood Sung, Yongxin Yang, Li Zhang, Tao Xiang, Philip HS Torr, and Timothy M Hospedales. Learning to compare: Relation network for few-shot learning. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 1199–1208, 2018.

[22] Oriol Vinyals, Charles Blundell, Timothy Lillicrap, Daan Wierstra, et al. Matching networks for one shot learning. *Advances in neural information processing systems*, 29:3630–3638, 2016.

[23] Adam Santoro, Sergey Bartunov, Matthew Botvinick, Daan Wierstra, and Timothy Lillicrap. Meta-learning with memory-augmented neural networks. In *International conference on machine learning*, pages 1842–1850. PMLR, 2016.

[24] Stephen Gould, Basura Fernando, Anoop Cherian, Peter Anderson, Rodrigo Santa Cruz, and Edison Guo. On differentiating parameterized argmin and argmax problems with application to bi-level optimization. *arXiv preprint arXiv:1607.05447*, 2016.

[25] Riccardo Grazzi, Luca Franceschi, Massimiliano Pontil, and Saverio Salzo. On the iteration complexity of hypergradient computation. In *International Conference on Machine Learning*, pages 3748–3758. PMLR, 2020.

[26] Kaiyi Ji, Junjie Yang, and Yingbin Liang. Bilevel optimization: Convergence analysis and enhanced design. In *International Conference on Machine Learning*, pages 4882–4892. PMLR, 2021.

[27] Saeed Ghadimi and Mengdi Wang. Approximation methods for bilevel programming. *arXiv preprint arXiv:1802.02246*, 2018.

[28] Mingyi Hong, Hoi-To Wai, Zhaoran Wang, and Zhuoran Yang. A two-timescale framework for bilevel optimization: Complexity analysis and application to actor-critic. *arXiv preprint arXiv:2007.05170*, 2020.

[29] Sijia Liu, Pin-Yu Chen, Xiangyi Chen, and Mingyi Hong. signsgd via zeroth-order oracle. In *International Conference on Learning Representations*, 2018.

[30] Saeed Ghadimi and Guanghui Lan. Stochastic first-and zeroth-order methods for nonconvex stochastic programming. *SIAM Journal on Optimization*, 23(4):2341–2368, 2013.

[31] Ian J Goodfellow, Jonathon Shlens, and Christian Szegedy. Explaining and harnessing adversarial examples. *arXiv preprint arXiv:1412.6572*, 2014.

[32] Aleksander Madry, Aleksandar Makelov, Ludwig Schmidt, Dimitris Tsipras, and Adrian Vladu. Towards deep learning models resistant to adversarial attacks. *arXiv preprint arXiv:1706.06083*, 2017.

[33] Boris N. Oreshkin, Pau Rodriguez, and Alexandre Lacoste. Tadam: Task dependent adaptive metric for improved few-shot learning, 2019.

[34] Alex Krizhevsky, Geoffrey Hinton, et al. Learning multiple layers of features from tiny images. 2009.

[35] Sachin Ravi and Hugo Larochelle. Optimization as a model for few-shot learning. 2016.

[36] Sébastien M. R. Arnold, Praateek Mahajan, Debajyoti Datta, Ian Bunner, and Konstantinos Saitas Zarkias. learn2learn: A library for meta-learning research, 2020. URL http://learn2learn.net.

A Hyperparameter Search

In this section, we provide more details on hyperparameter tuning for the lower-level learning rate. For Sign-MAML, the search range is $[0.0035, 0.005, 0.0065, 0.0075, 0.01]$; for FO-MAML, the search range is $[0.06, 0.08, 0.1, 0.12, 0.14, 0.16]$; for MAML, the search range is $[0.06, 0.08, 0.1, 0.12, 0.14, 0.16]$. If the initial optimal learning rate happens at the end of the range, we continue search in that direction until we find a better one that is within range. For example, if the optimal learning rate initially found is 0.16, then we may search 0.18 and 0.2. If 0.18 outperforms 0.16 and 0.2, we stop at this point; if 0.2 outperforms the other two, we repeat the process and search further.

B Train Loss

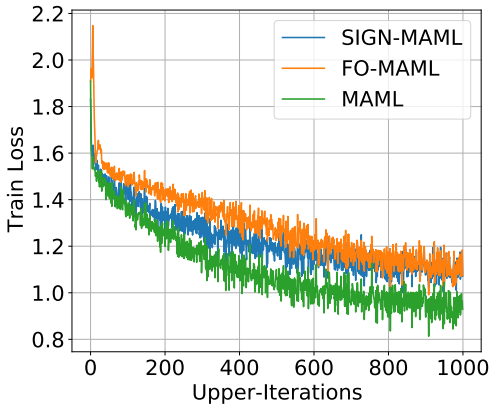
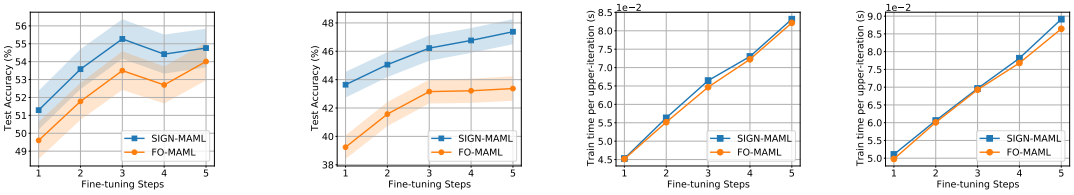


Figure 3: Train loss for MiniImageNet 5-way 5-shot classification. The lower-level learning rates for Sign-MAML, FO-MAML and MAML are 0.005, 0.06 and 0.06 respectively. Meta-batch size is 32.

C Additional results on MiniImageNet for different fine-tuning steps



(a) Accuracy 5-way 2-shot (b) Accuracy 7-way 2-shot (c) Time 5-way 2-shot (d) Time 7-way 2-shot

Figure 4: MiniImageNet classification against the choice of the number of fine-tuning steps: (a) 5-way 2-shot classification accuracy, (b) 7-way 2-shot classification accuracy, (c) 5-way 2-shot computation time and (d) 7-way 2-shot computation time.