GAML: GEOMETRY-AWARE META-LEARNING VIA A FULLY ADAPTIVE PRECONDITIONER

Anonymous authors
Paper under double-blind review

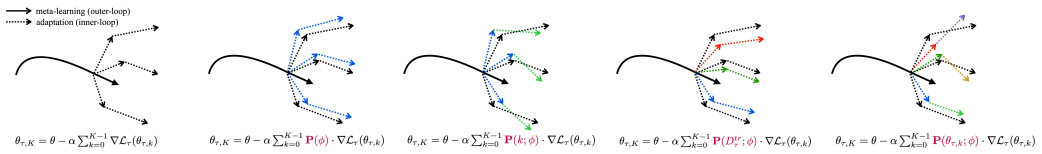
ABSTRACT

Model-Agnostic Meta-Learning (MAML) is one of the most successful meta-learning algorithms. It has a bi-level optimization structure, where the outer-loop process learns the shared initialization and the inner-loop process optimizes the task-specific weights. Although MAML relies on the standard gradient descent in the inner-loop, recent works have shown that it can be beneficial to control the inner-loop's gradient descent with a meta-learned preconditioner. The existing preconditioners, however, cannot adapt in a task-specific and path-dependent way at the same time. Also, most of them do not consider the geometry of the loss surface. In this work, we propose Geometry-Aware Meta-Learning (GAML) that can overcome the limitations. GAML can efficiently meta-learn a preconditioner that is dependent on the task-specific parameters and its preconditioner can be shown to be a Riemannian metric that defines the geometry of the loss surface. Therefore, we can perform a fully-adaptive and geometry-aware optimization in the inner-loop. Experiment results show that GAML outperforms the state-of-the-art MAML family and PGD-MAML family for a variety of few-shot learning tasks.

1 INTRODUCTION

Human can quickly learn new concepts with only a small number of samples by exploiting the past experience. On the contrary, modern deep neural networks still require a large number of labeled samples and a large computational resource to learn a new concept. Meta-learning, or *learning to learn*, addresses this issue by extracting prior-knowledge known as meta-knowledge from a variety of tasks and by improving the generalization capability over the new tasks(Andrychowicz et al., 2016; Vinyals et al., 2016; Munkhdalai & Yu, 2017; Snell et al., 2017; Nichol & Schulman, 2018; Zintgraf et al., 2019; Rajeswaran et al., 2019; Achille et al., 2019). Among the meta-learning algorithms, the category of optimization-based meta-learning (Finn et al., 2017; 2018; Raghu et al., 2019; Baik et al., 2021; Ding et al., 2022) has been gaining popularity due to its flexible applicability over diverse fields including robotics (Song et al., 2020; Wen et al., 2021), medical image analysis (Maicas et al., 2018; Singh et al., 2021), language modeling (Mi et al., 2019; Liu et al., 2020), and object detection (Wu et al., 2020; Perez-Rua et al., 2020). In particular, Model-Agnostic Meta-Learning (MAML) is one of the most popular optimization-based algorithms.

Many of the recent works have improved MAML by adopting a Preconditioned Gradient Descent (PGD) as the inner-loop optimization (Li et al., 2017; Lee & Choi, 2018; Park & Oliva, 2019; Simon et al., 2020; Rajasegaran et al., 2020; Zhao et al., 2020; Von Oswald et al., 2021). We will address the PGD-based MAML algorithms as *PGD-MAML family*. PGD is different from the ordinary gradient descent because it performs a preconditioning on the gradient using a preconditioning matrix $\mathbf{P}$, also called a *preconditioner*. A PGD-MAML algorithm meta-learns not only the initialization parameter θ_0 of the network but also the meta-parameter ϕ of the preconditioner $\mathbf{P}$. For the inner-loop optimization, $\mathbf{P}$ is kept static in most of the previous works (Figure 1(b)) (Li et al., 2017; Lee & Choi, 2018; Park & Oliva, 2019; Zhao et al., 2020; Von Oswald et al., 2021). Some of the previous works considered adapting the preconditioner $\mathbf{P}$ with the inner-step k (Figure 1(c)) (Rajasegaran et al., 2020) and some others with the individual task (Figure 1(d)) (Simon et al., 2020). None of the existing works, however, successfully came up with a PGD that can perform a full adaptation as shown in Figure 1(e).



(a) MAML (b) Non-adaptive $\mathbf{P}(\phi)$ (c) Adaptive $\mathbf{P}(k;\phi)$ (d) Adaptive $\mathbf{P}(D_\tau^{tr};\phi)$ (e) Adaptive $\mathbf{P}(\theta_{\tau,k};\phi)$

Figure 1: Diagrams of MAML and PGD-MAML family. For the inner-loop adaptation in each diagram, the dotted lines of the same color indicate that they use a common preconditioning matrix (preconditioner). (a) MAML adaptation: no preconditioner is used (i.e., $\mathbf{P} = \mathbf{I}$). (b) $\mathbf{P}(\phi)$: a constant preconditioner is used in the inner-loop where the preconditioner's meta-parameter ϕ is meta-learned. (c) $\mathbf{P}(k;\phi)$: a constant preconditioner is used for each inner-step k. Preconditioner for each step is meta-learned, but $\mathbf{P}(k,\phi)$ is not task-specific. (d) $\mathbf{P}(D_\tau^{tr};\phi)$: a constant preconditioner is used for each task. Preconditioner for each task is meta-learned, but $\mathbf{P}(D_\tau^{tr};\phi)$ is not dependent on k. (e) GAML adapts $\mathbf{P}(\theta_{\tau,k};\phi)$: a fully adaptive preconditioner is used where it is *task-specific* and *path-dependent*. Instead of saying 'dependent on k', we specifically say it is *path-dependent* because the exact dependency is on the task-specific parameter set $\theta_{\tau,k}$ that is much more informative than k.

In this study, we propose a new PGD method named **G**eometry **A**ware **M**eta-**L**earning (GAML). It is a *fully adaptive* preconditioner that is aware of the geometry of the loss surface. To be precise, GAML satisfies the following two desirable properties. First, GAML's preconditioner $\mathbf{P}_{\text{GAML}}$ is a fully adaptive preconditioner that can adapt to the individual task (*task-specific*) and to the optimization-path (*path-dependent*). The full adaptation is made possible by having the preconditioner depend on the task-specific parameter $\theta_{\tau,k}$. The extended level of adaptation implies that it might be possible to achieve a high-performance over a broad range of applications. Second, we prove that $\mathbf{P}_{\text{GAML}}$ is a Riemannian metric (Amari, 1967; 1996; 1998; Amari & Douglas, 1998; Kakade, 2001). When a parameter space has a certain underlying structure, the ordinary gradient of a function does not represent its steepest direction (Amari, 1998). A preconditioning can be viewed as changing the geometry of the parameter space (Himmelblau et al., 2018). When the preconditioning matrix is a Riemannian metric, it defines a corresponding geometry of the underlying structure and enables an efficient steepest descent. Among the existing PGD-MAML works, Lee & Choi (2018) show that their algorithm implements a constant positive definite metric as in Figure 1(b). While this is often understood as a Riemannian metric, the algorithm is not a strict preconditioner because it performs a modification in the network structure by inserting linear layers. A preconditioner is defined as a matrix that is used for filtering the gradient only, and it should not affect the feedforward calculation itself. In contrast to the algorithm, GAML is a strict preconditioner and thus it is completely model-agnostic.

To come up with a fully adaptive preconditioner that is a Riemannian metric, we utilize SVD (Singular Value Decomposition) operation in our design. While the SVD operation does not incur a large computational burden for the benchmark architectures, it can become a burden for large-scale architectures. To resolve this potential problem, we propose a low-computation strategy that can be connected to a theoretical approximation of GAML.

To demonstrate the effectiveness of GAML, we empirically evaluate our algorithm on popular few-shot learning tasks; few-shot regression, few-shot classification, and few-shot cross-domain classification. The results show that GAML outperforms the state-of-the-art MAML family and the state-of-the-art PGD-MAML family. For example, GAML outperforms MAML by 6.97% and 5.90% in 5-way 1-shot settings over the mini-ImageNet and tiered-ImageNet datasets, respectively.

The **main contributions** of our work can be summarized as follows:

- We propose a new preconditioned gradient descent method called GAML. It can learn a fully adaptive preconditioner that is aware of the loss geometry in inner-loop optimization.
- We prove that GAML's preconditioner has two desirable properties: (1) It depends on task-specific parameter $\theta_{\tau,k}$. (2) It is a Riemannian metric that is model-agnostic.
- For large-scale architectures, we derive an approximated GAML method that can retain most of the performance benefits while the computational burden is kept similar to MAML's.
- In popular few-shot learning tasks, GAML outperforms the state-of-the-art MAML family and PGD-MAML family.

2 BACKGROUND

2.1 MODEL-AGNOSTIC META-LEARNING (MAML)

The goal of MAML (Finn et al., 2017) is to find the best initialization that the model can quickly adapt from, such that the model can perform well for a new task. MAML consists of two levels of main optimization processes: inner-loop optimization and outer-loop optimization. Consider the model $f_\theta(\cdot)$ with parameter θ. For a task $\tau = \{D_\tau^{tr}, D_\tau^{val}\}$ sampled from the task distribution $p(\mathcal{T})$, the inner-loop optimization is defined as:

$$\theta_{\tau,K} = \theta_{\tau,0} - \alpha \sum_{k=0}^{K-1} \nabla_{\theta_{\tau,k}} \mathcal{L}_\tau^{in}(\theta_{\tau,k}; D_\tau^{tr}) \;\; s.t \;\; \theta_{\tau,0} = \theta, \tag{1}$$

where $\theta_{\tau,k}$ is task-specific parameters for task τ, α is the learning rate of inner-loop optimization, $\mathcal{L}_\tau^{in}$ is the inner-loop's loss function, and K is the number of gradient descent steps. With $D_{\tau_i}^{val}$ in each task, we can define outer-loop optimization as:

$$\theta \leftarrow \theta - \beta \nabla_\theta \mathbb{E}_\tau \left[\mathcal{L}_\tau^{out}(\theta_{\tau,K}; D_\tau^{val}) \right], \tag{2}$$

where β is the learning rate for outer-loop optimization, and $\mathcal{L}_\tau^{out}$ is the outer-loop's loss function.

2.2 UNFOLDING: RESHAPING A TENSOR INTO A MATRIX

In our work, the concept of *unfolding* is used to transform the gradient tensor of convolutional kernels into a matrix form. *Tensor unfolding*, also known as matricization or flattening, is the process of reshaping the elements of an N-dimensional tensor $\mathbf{X} \in \mathbb{R}^{I_1 \times \cdots \times I_N}$ into a matrix (Kolda & Bader, 2009). The mode-n unfolding of an N-dimensional tensor $\mathbf{X} \in \mathbb{R}^{I_1 \times \cdots \times I_N}$ is defined as:

$$\mathbf{X} \xrightarrow[\text{mode-n unfolding}]{} \mathbf{X}_{[n]} \in \mathbb{R}^{I_n \times I_M}, \text{ where } I_M = \prod_{k \neq n} I_k \tag{3}$$

For example, the weight tensor of a convolutional layer is represented as a 4-D tensor ($\mathbf{W} \in \mathbb{R}^{C_{out} \times C_{in} \times k_h \times k_w}$) where it is composed of kernels and it can be unfolded into a matrix as one of the following four forms: (1) $\mathbf{W}_{[1]} \in \mathbb{R}^{C_{out} \times (C_{in} k_h k_w)}$, (2) $\mathbf{W}_{[2]} \in \mathbb{R}^{C_{in} \times (C_{out} k_h k_w)}$, (3) $\mathbf{W}_{[3]} \in \mathbb{R}^{k_h \times (C_{out} C_{in} k_w)}$, (4) $\mathbf{W}_{[4]} \in \mathbb{R}^{k_w \times (C_{out} C_{in} k_h)}$. Following the previous works (Wen et al., 2017; Li & Shi, 2018; Xu et al., 2019) where mode-1 unfolding was adopted, we also apply mode-1 unfolding in our algorithm.

2.3 PRECONDITIONED GRADIENT DESCENT (PGD)

Preconditioned Gradient Descent (PGD) is a method that minimizes the empirical risk through a gradient update with a preconditioner that modifies the geometry of the loss surface. Given model parameters θ and task $\tau = \{D_\tau^{tr}, D_\tau^{val}\}$, we can define the preconditioned gradient update with a preconditioner $\mathbf{P}$ as the following:

$$\theta_{\tau,k+1} = \theta_{\tau,k} - \alpha \mathbf{P} \nabla_{\theta_{\tau,k}} \mathcal{L}_\tau(\theta_{\tau,k}; D_\tau^{tr}), \quad k = 0, 1, \cdots \text{ and } \theta_{\tau,0} = \theta, \tag{4}$$

where $\mathcal{L}_\tau(\theta_{\tau,k}; D_\tau^{tr})$ is the empirical loss for task τ and parameters $\theta_{\tau,k}$. Setting $\mathbf{P} = \mathbf{I}$ recovers Eq. (4) of the basic gradient descent (GD). Choice of $\mathbf{P}$ for exploiting the second-order information includes the inverse Fisher information matrix $\mathbf{F}^{-1}$ which leads to the natural gradient descent (NGD) (Amari, 1998); the inverse Hessian $\mathbf{H}^{-1}$ which corresponds to the Newton's method (LeCun et al., 2012); and the diagonal matrix estimation with the past gradients which results in the adaptive gradient methods (Duchi et al., 2011; Kingma & Ba, 2014). These preconditioners regulate the geometry of the loss surface, often reducing the effect of pathological curvature and speeding up the optimization (Amari et al., 2020).

3 METHODOLOGY

In this section, we propose a new preconditioned gradient descent method called GAML for a flexible and effective geometry-aware adaptation in the MAML framework. In Section 3.1, we will

introduce GAML process in the inner-loop optimization and describe how to meta-train GAML with an outer-loop optimization. In Section 3.2, we will prove that GAML has two desirable properties: (1) a preconditioner induced by GAML depends on task-specific parameters; and (2) it is a Riemannian metric. In Section 3.3, we will provide a theoretically grounded method for approximating GAML such that computational efficiency can be maintained ever for large-scale architectures.

3.1 GAML: GEOMETRY-AWARE META-LEARNING WITH A FULLY ADAPTIVE PRECONDITIONER

3.1.1 INNER-LOOP OPTIMIZATION

We consider an L-layer neural network $f_\theta(\cdot)$ with parameters $\theta = \{\mathbf{W}^1, \cdots, \mathbf{W}^l, \cdots, \mathbf{W}^L\}$. In the standard MAML with task $\tau \sim p(\mathcal{T})$, each $\mathbf{W}^l$ is adapted with the gradient update as below:

$$\mathbf{W}^l_{\tau,K} \leftarrow \mathbf{W}^l_{\tau,0} - \alpha \cdot \sum_{k=0}^{K-1} \mathbf{G}^l_{\tau,k} \;\; s.t \;\; \mathbf{W}^l_{\tau,0} = \mathbf{W}^l, \tag{5}$$

where $\mathbf{G}^l_{\tau,k} = \nabla_{\mathbf{W}^l_{\tau,k}} \mathcal{L}^{in}_\tau(\theta_{\tau,k}; D^{tr}_\tau)$ is the gradient with respect to $\mathbf{W}^l_{\tau,k}$.

In GAML, we first use the mode-1 unfolding to reshape the gradient tensor into a matrix form (see Section 2.2). For a convolutional layer (i.e., $\mathbf{G}^l_{\tau,k} \in \mathbb{R}^{C_{out} \times C_{in} \times k \times k}$), we reshape the gradient tensor as below:

$$\mathbf{G}^l_{\tau,k} \xrightarrow[\text{mode-1 unfolding}]{} \begin{cases} \mathbf{G}^l_{\tau,k} \in \mathbb{R}^{C_{out} \times C_{in}k^2} & \text{if } C_{out} \leq C_{in}k^2 \\ \mathbf{G}^l_{\tau,k} \in \mathbb{R}^{C_{in}k^2 \times C_{out}} & \text{if } C_{in}k^2 < C_{out}, \end{cases} \tag{6}$$

where $\mathbf{G}^l_{\tau,k}$ denotes $(\mathbf{G}^l_{\tau,k})_{[1]}$ for the notational brevity. For a linear layer in a matrix form, there is no need for an unfolding.

Second, we transform the singular values of the gradient matrix using meta parameters $\phi = \{\mathbf{M}^1, \cdots, \mathbf{M}^l, \cdots, \mathbf{M}^L\} = \{\mathbf{M}^l\}^L_{l=1}$. We will use ϕ and $\{\mathbf{M}^l\}^L_{l=1}$ interchangeably. The meta parameters $\{\mathbf{M}^l\}^L_{l=1}$ are diagonal matrices with positive elements, and they are applied to the gradient matrix as below:

$$\tilde{\mathbf{G}}^l_{\tau,k} = \mathbf{U}^l_{\tau,k}(\mathbf{M}^l \cdot \mathbf{\Sigma}^l_{\tau,k}){\mathbf{V}^l_{\tau,k}}^T, \tag{7}$$

where $\mathbf{G}^l_{\tau,k} = \mathbf{U}^l_{\tau,k}\mathbf{\Sigma}^l_{\tau,k}{\mathbf{V}^l_{\tau,k}}^T$ is the singular value decomposition (SVD) of $\mathbf{G}^l_{\tau,k}$.

Finally, we reshape $\tilde{\mathbf{G}}^l_{\tau,k}$ back to its original gradient tensor form $\tilde{\mathbf{G}}^l_{\tau,k}$. The resulting preconditioned gradient descent of GAML becomes the following:

$$\mathbf{W}^l_{\tau,K} \leftarrow \mathbf{W}^l_{\tau,0} - \alpha \cdot \sum_{k=0}^{K-1} \tilde{\mathbf{G}}^l_{\tau,k} \;\; s.t \;\; \mathbf{W}^l_{\tau,0} = \mathbf{W}^l, \tag{8}$$

where $\tilde{\mathbf{G}}^l_{\tau,k}$ is the preconditioned gradient based on the meta parameters ϕ.

3.1.2 OUTER-LOOP OPTIMIZATION

For meta-learning, GAML follows the typical MAML outer-loop process. Unlike MAML, however, GAML meta-learns two meta parameter sets θ and ϕ as follows:

$$\theta \leftarrow \theta - \beta_1 \nabla_\theta \mathbb{E}_\tau \left[\mathcal{L}^{out}_\tau(\theta_{\tau,K}; D^{val}_\tau)\right], \tag{9}$$

$$\phi \leftarrow \phi - \beta_2 \nabla_\phi \mathbb{E}_\tau \left[\mathcal{L}^{out}_\tau(\theta_{\tau,K}; D^{val}_\tau)\right], \tag{10}$$

where β_1 and β_2 are the learning rates for the outer-loop optimization. To minimize the influence of the meta parameters $\phi = \{\mathbf{M}^l\}^L_{l=1}$ at the beginning of training, we initialize $\mathbf{M}^l$ as an identity matrix for all l. We use the ADAM optimizer for outer-loop optimization and simultaneously update θ and ϕ. The training procedure is provided in Algorithm 1.

Algorithm 1 Geometry-aware meta-learning (GAML)

Require: $p(\mathcal{T})$: distribution over tasks
Require: α, β_1, β_2: learning rates
Randomly initialize $\theta = \{\mathbf{W}^1, \cdots, \mathbf{W}^L\}$ and initialize $\phi = \{\mathbf{M}^1, \cdots, \mathbf{M}^L\}$ as $\mathbf{M}^l = \mathbf{I}$
while not converged **do**
  Sample a batch of tasks $\mathcal{T}_i \sim p(\mathcal{T})$
  for all $\tau \in \mathcal{T}_i$ **do**
    for inner-loop step $k = 0$ **to** $K - 1$ **do**
      for layer $l = 1$ **to** L **do**
        Compute $\mathbf{G}^l_{\tau,k} = \nabla_{\mathbf{W}^l_{\tau,k}} \mathcal{L}^{in}_\tau(\theta_{\tau,k}; D^{tr}_\tau)$ using D^{tr}_τ
        Reshape $\mathbf{G}^l_{\tau,k}$ to $\mathbf{G}^l_{\tau,k}$ via Eq. (6)
        Transform $\mathbf{G}^l_{\tau,k}$ to $\tilde{\mathbf{G}}^l_{\tau,k}$ using $\mathbf{M}^l$ via Eq. (7)
        Reshape $\tilde{\mathbf{G}}^l_{\tau,k}$ back to the original form of gradient tensor as $\tilde{\mathbf{G}}^l_{\tau,k}$
        Compute l-layer adapted weight: $\mathbf{W}^l_{\tau,k+1} = \mathbf{W}^l_{\tau,k} - \alpha \cdot \tilde{\mathbf{G}}^l_{\tau,k}$
      end for
    end for
    Compute $\mathcal{L}^{out}_\tau(\theta_{\tau,K}; D^{val}_\tau)$ by evaluating $\mathcal{L}^{out}_\tau$ w.r.t D^{val}_τ.
  end for
  Update the weights and meta parameters:
  $\theta \leftarrow \theta - \beta_1 \nabla_\theta \sum_{\tau \in \mathcal{T}} \mathcal{L}^{out}_\tau(\theta_{\tau,K}; D^{val}_\tau)$
  $\phi \leftarrow \phi - \beta_2 \nabla_\phi \sum_{\tau \in \mathcal{T}} \mathcal{L}^{out}_\tau(\theta_{\tau,K}; D^{val}_\tau)$
end while

3.2 THE DESIRABLE PROPERTIES OF GAML

In this section, we prove that GAML's preconditioner $\mathbf{P}_{\text{GAML}}$ satisfy two desirable properties.

Theorem 1. *Let $\tilde{\mathbf{G}}^l_{\tau,k} \in \mathbb{R}^{m \times n}$ be the 'l-layer k-th inner-step gradient' transformed by meta parameter $\mathbf{M}^l$ for task τ. Then preconditioner $\mathbf{P}_{GAML}$ induced by $\tilde{\mathbf{G}}^l_{\tau,k}$ is a Riemannian metric and depends on the task-specific parameter $\theta_{\tau,k}$.*

The proof of Theorem 1 is provided in Supplementary B. The two properties are explained below.

Property 1. Dependency on task-specific parameters: Theorem 1 formally shows that $\mathbf{P}_{\text{GAML}}$ depends on the task-specific parameters $\theta_{\tau,k}$. While the previous works considered a non-adaptive preconditioner $\mathbf{P}(\phi)$ (Li et al., 2017; Park & Oliva, 2019; Von Oswald et al., 2021; Lee & Choi, 2018; Zhao et al., 2020; Rajasegaran et al., 2020) and a partially adaptive preconditioner $\mathbf{P}(k; \phi)$ (Rajasegaran et al., 2020) or $\mathbf{P}(D^{tr}_\tau; \phi)$ (Simon et al., 2020), GAML is the most advanced adaptive preconditioner in that it is fully adaptive by being dependent on $\theta_{\tau,k}$. As shown in Figure 1, GAML is the only preconditioner that is task-specific and path-dependent (i.e., dependent on the inner loop optimization path).

Property 2. Riemannian metric: Preconditioning can be interpreted as modifying the geometry of the parameter space (Himmelblau et al., 2018). If the modified parameter space has a certain underlying structure, the ordinary gradient of a function $\nabla\mathcal{L}$ does not represent its steepest direction (Amari, 1998). To define the steepest direction, we need a Riemannian metric G, which is a positive-definite and smoothly-varying matrix (Amari, 1998). The metric defines the steepest descent direction by $-G^{-1}\nabla\mathcal{L}$ (Amari, 1998). If a preconditioning matrix is a Riemannian metric, it defines a geometry of the underlying structure and enables an efficient steepest descent. To achieve this, we prove that $\mathbf{P}_{\text{GAML}}$ is a Riemannian metric in Theorem 1. Our metric consists of two factors, a unitary matrix of the inner-loop gradient $\mathbf{U}_{\tau,k}$ and a meta-parameter $\mathbf{M}$. $\mathbf{M}$ allows us to reflect the shared geometry information across the tasks. In addition, task-specific and path-dependent geometry information can be reflected in the metric through $\mathbf{U}_{\tau,k}$.

3.3 APPROXIMATION OF GAML

Hu et al. (2022) emphasized that the use of a large architecture is an important factor for improving the performance of meta-learning. Indeed, many of the recent works have achieved a state-of-the-art performance with a large network (Reed et al., 2022; Hu et al., 2022; Melo, 2022; Nguyen & Grover, 2022; Doersch et al., 2020; Bommasani et al., 2021). In the case of GAML, it utilizes SVD operation and its computation can increases in $O(s^3)$ where s is the scaling factor of the network size. This

Table 1: Few-shot regression for the sinusoid regression benchmark with a *2-layer MLP* backbone. We report MSE $\pm$ 95% confidence intervals(ci) for 600 tasks following the setup in (Finn et al., 2017). † denotes PGD-MAML family.

Algorithm	5-shot	10-shot	20-shot
MAML (Finn et al., 2017)	1.13 ± 0.18	0.77 ± 0.11	0.48 ± 0.08
Meta-SGD† (Li et al., 2017)	0.90 ± 0.16	0.53 ± 0.09	0.31 ± 0.05
MT-Net† (Lee & Choi, 2018)	0.76 ± 0.09	0.49 ± 0.05	0.33 ± 0.04
ALFA (Baik et al., 2020a)	0.92 ± 0.19	0.62 ± 0.16	0.34 ± 0.07
L2F (Baik et al., 2020b)	$0.71 \pm$ N/A	$0.37 \pm$ N/A	$0.16 \pm$ N/A
PAMELA† (Rajasegaran et al., 2020)	0.54 ± 0.06	0.41 ± 0.04	0.17 ± 0.03
MeTAL (Baik et al., 2021)	0.74 ± 0.18	0.44 ± 0.11	0.21 ± 0.06
GAML†	$\mathbf{0.16 \pm 0.04}$	$\mathbf{0.04 \pm 0.01}$	$\mathbf{0.01 \pm 0.01}$

issue, however, can be significantly alleviated with an approximation strategy that completely avoids the SVD operation. We provide a theorem for the approximation strategy below.

Theorem 2. *Let $\mathbf{G} \in \mathbb{R}^{m \times n}$ be a gradient tensor reshaped into a matrix form. As n becomes large, GAML's preconditioned gradient matrix $\tilde{\mathbf{G}}$ asymptotically becomes equivalent to $\mathbf{MG}$, where $\mathbf{M}$ is a diagonal matrix. Therefore, the following holds for a large n.*

$$\tilde{\mathbf{G}} \cong \mathbf{MG} \tag{11}$$

The proof can be found in Supplementary B and it shows that we can use the approximation $\tilde{\mathbf{G}}^l_{\tau,k} = \mathbf{M}^l \cdot \mathbf{G}^l_{\tau,k}$ instead of Eq. (7) when n is large. Note that we have chosen the larger dimension of the gradient matrix as n when reshaping with Eq. (6). By removing the SVD operation, the computational burden becomes about the same for the approximated GAML and MAML because GAML has about the same number of parameters as MAML as will be shown in Table 5.

4 Experiments

In this section, we show the superiority of GAML by comparing it with the state-of-the-art PGD-MAML family and the MAML family. We evaluate GAML using the widely used benchmarks: few-shot regression, few-shot classification, and cross-domain adaptation. Hyper-parameters setups used in our experiments are provided in the Supplementary A.

4.1 Few-shot regression

Datasets and experimental setup. The goal of few-shot regression is to fit an unknown target function for the given K sample points from the function. For the evaluation of few-shot regression, we use the sinusoid regression benchmark (Finn et al., 2017). In this benchmark, sinusoid is used as the target function. Each task has a sinusoid $y(x) = A\sin(\omega x + b)$ as the target function, where the parameter values are within the following range: amplitude $A \in [0.1, 5.0]$, frequency $\omega \in [0.8, 1.2]$, and phase $b \in [0, \pi]$. For each task, input data point x is sampled from $[-5.0, 5.0]$. In the experiment, we use a simple Multi-Layer Perceptron (MLP) with 1-dimensional input/output and 40-dimensional hidden layers (2 hidden layers), following the setting in Finn et al. (2017).

Results. We evaluate GAML and compare it with MAML and PGD-MAML family on a regression task using the performance metric of Mean Squared Error (MSE). As shown in Table 1, GAML consistently achieves the lowest MSE scores with the lowest confidence intervals in all three cases. The MSE score decreases drastically as the number of shots is increased. The performance of GAML is improved by 89% on 10-shot and 94% on 20-shot compared to the previous state-of-the-art performance.

4.2 Few-shot classification

Datasets and experimental setup. For the few-shot classification, we evaluate two benchmarks: (1) mini-ImageNet (Vinyals et al., 2016): This dataset has 100 classes and it is a subset of ImageNet (Deng et al., 2009). We use the same split as in Ravi & Larochelle (2016), with 64, 16 and 20 classes for train, validation and test. (2) Tiered-ImageNet (Ren et al., 2018): This is also a subset of ImageNet

Table 2: 5-way few-shot classification accuracy (%) on mini-ImageNet with a *Conv-4* backbone. We report mean ± 95% confidence intervals(ci) for 600 tasks according to Finn et al. (2017). † denotes PGD-MAML family. The full comparison with additional works can be found in Supplementary Table C.2.

Algorithm	5-way 1-shot	5-way 5-shot
MAML (Finn et al., 2017)	47.89 ± 1.20	64.59 ± 0.88
Meta-SGD† (Li et al., 2017)	50.47 ± 1.87	64.00 ± 0.90
LLAMA (Grant et al., 2018).	49.40 ± 1.83	N/A
T-net† (Lee & Choi, 2018)	50.86 ± 1.82	N/A
MT-net† (Lee & Choi, 2018)	51.70 ± 1.84	N/A
BMAML (Yoon et al., 2018)	53.80 ± 1.46	64.23 ± 0.69
iMAML-HF (Rajeswaran et al., 2019)	49.30 ± 1.88	N/A
WarpGrad† (Flennerhag et al., 2019)	52.30 ± 0.90	68.40 ± 0.60
MC† (Park & Oliva, 2019)	54.08 ± 0.88	67.99 ± 0.73
MH† (Zhao et al., 2020)	49.41 ± 0.96	67.16 ± 0.42
ALFA (Baik et al., 2020a)	50.58 ± 0.51	69.12 ± 0.47
ModGrad† (Simon et al., 2020)	53.20 ± 0.86	69.17 ± 0.69
PAMELA† (Rajasegaran et al., 2020)	53.50 ± 0.89	70.51 ± 0.67
SignMAML (Fan et al., 2021)	42.90 ± 1.50	60.70 ± 0.70
Sparse-MAML† (Von Oswald et al., 2021)	50.35 ± 0.39	67.03 ± 0.74
Sparse-ReLU-MAML† (Von Oswald et al., 2021)	50.39 ± 0.89	64.84 ± 0.46
Sparse-MAML+† (Von Oswald et al., 2021)	51.04 ± 0.59	67.03 ± 0.74
MeTAL (Baik et al., 2021)	52.63 ± 0.37	70.52 ± 0.29
Sharp-MAML (Abbas et al., 2022)	$50.28 \pm$ N/A	$65.04 \pm$ N/A
FBM (Yang et al., 2022)	50.62 ± 1.79	64.78 ± 0.35
CxGrad (Lee et al., 2022)	51.80 ± 0.46	69.82 ± 0.42
HyperMAML (Przewięźlikowski et al., 2022)	51.84 ± 0.57	66.29 ± 0.43
EEML (Li et al., 2022)	52.42 ± 1.75	68.40 ± 0.95
GAML (approx.)†	53.52 ± 0.88	70.75 ± 0.67
GAML†	$\mathbf{54.86 \pm 0.85}$	$\mathbf{71.55 \pm 0.61}$

Table 3: 5-way few-shot classification accuracy (%) on Tiered-ImageNet with a *Conv-4* backbone. We report mean ± 95% confidence intervals(ci) for 600 tasks according to Finn et al. (2017). † denotes PGD-MAML family.

Algorithm	5-way 1-shot	5-way 5-shot
Meta-SGD† (Li et al., 2017)	50.92 ± 0.93	69.28 ± 0.80
MAML (Finn et al., 2017)	51.70 ± 1.80	70.30 ± 1.80
MT-net† (Lee & Choi, 2018)	51.95 ± 1.83	N/A
WarpGrad† (Flennerhag et al., 2019)	57.20 ± 0.90	74.10 ± 0.70
BOIL (Oh et al., 2020).	48.58 ± 0.27	69.37 ± 0.12
ALFA (Baik et al., 2020a)	53.16 ± 0.49	70.54 ± 0.46
L2F (Baik et al., 2020b)	54.40 ± 0.50	73.34 ± 0.44
ARML (Yao et al., 2020)	52.91 ± 1.83	N/A
PAMELA† (Rajasegaran et al., 2020)	54.81 ± 0.88	74.39 ± 0.71
Sparse-ReLU-MAML† (Von Oswald et al., 2021)	53.18 ± 0.52	69.06 ± 0.28
Sparse-MAML† (Von Oswald et al., 2021)	53.47 ± 0.53	68.83 ± 0.65
Sparse-MAML+† (Von Oswald et al., 2021)	53.91 ± 0.67	69.92 ± 0.21
MeTAL (Baik et al., 2021)	54.34 ± 0.31	70.40 ± 0.21
CxGrad (Lee et al., 2022)	55.55 ± 0.46	73.55 ± 0.41
ECML (Hiller et al., 2022)	47.34 ± 0.88	64.77 ± 0.75
GAML (approx.)†	56.86 ± 0.91	74.41 ± 0.72
GAML†	$\mathbf{57.60 \pm 0.93}$	$\mathbf{74.90 \pm 0.68}$

with 608 classes grouped into 34 high-level categories, divided into 20, 6 and 8 for train, validation, and test. For all the experiments, our model follows the standard *Conv-4* backbone used in (Vinyals et al., 2016), comprising of 4 modules with 3×3 convolutions with 128 filters followed by batch normalization (Ioffe & Szegedy, 2015), ReLU non-linearity, and 2×2 max-pooling. Following the experimental protocol in (Finn et al., 2017), we use 15 samples per class in the query set to compute the meta gradients. In meta training and meta testing, the inner-loop optimization is updated in five steps and ten steps, respectively.

Results. Table 2 & 3 present the performance of GAML, state-of-the-art PGD-MAML family, and state-of-the-art MAML-family on mini-ImageNet and Tiered-ImageNet under two typical settings: 5-way 1-shot and 5-way 5-shot. GAML outperforms all of the previous PGD-MAML and MAML family. Compared to the state-of-the-art MAML family, GAML improves the performance with a quite significant margin for both mini-ImageNet and Tiered-ImageNet. Compared to the state-

Table 4: 5-way few-shot cross domain classification accuracy (%) with a *Conv-4* backbone, meta training on mini-ImageNet, and meta-testing on Tiered-ImageNet, CUB, or Cars. We report mean $\pm$ 95% confidence intervals(ci) for 600 tasks according to Finn et al. (2017). † denotes PGD-MAML family.

	Tiered-ImageNet		CUB		Cars	
Algorithm	1-shot	5-shot	1-shot	5-shot	1-shot	5-shot
MAML	51.61 ± 0.20	65.76 ± 0.27	40.51 ± 0.08	53.09 ± 0.16	33.57 ± 0.14	44.56 ± 0.21
ANIL	52.82 ± 0.29	66.52 ± 0.28	41.12 ± 0.15	55.82 ± 0.21	34.77 ± 0.31	46.55 ± 0.29
BOIL	53.23 ± 0.41	69.37 ± 0.23	44.20 ± 0.15	60.92 ± 0.11	36.12 ± 0.29	50.64 ± 0.22
BMAML	N/A	N/A	33.52 ± 0.36	51.35 ± 0.16	N/A	N/A
ALFA	N/A	N/A	N/A	58.35 ± 0.25	N/A	N/A
L2F	N/A	N/A	N/A	60.89 ± 0.22	N/A	N/A
MeTAL	N/A	N/A	N/A	58.20 ± 0.24	N/A	N/A
HyperMAML	N/A	N/A	36.52 ± 0.61	49.43 ± 0.14	N/A	N/A
CxGrad	N/A	N/A	N/A	63.92 ± 0.44	N/A	N/A
Sparse-MAML†	53.47 ± 0.53	68.83 ± 0.65	41.37 ± 0.73	60.58 ± 1.10	35.90 ± 0.50	52.63 ± 0.56
Sparse-ReLU-MAML†	53.77 ± 0.94	68.12 ± 0.69	42.89 ± 0.45	57.53 ± 0.94	36.04 ± 0.55	49.95 ± 0.42
Sparse-MAML+†	53.91 ± 0.67	69.92 ± 0.21	43.43 ± 1.04	62.02 ± 0.78	37.14 ± 0.77	53.18 ± 0.44
GAML†	$\mathbf{58.56 \pm 0.93}$	$\mathbf{72.82 \pm 0.77}$	$\mathbf{44.74 \pm 0.75}$	$\mathbf{64.88 \pm 0.72}$	$\mathbf{38.44 \pm 0.77}$	$\mathbf{55.04 \pm 0.77}$

of-the-art PGD-MAML family, GAML shows that the 1- and 5-shot accuracy can be increased by 1.4 % and 1.5 % on mini-ImageNet, and by 0.7 % and 0.68 % on Tiered-ImageNet, respectively. We also evaluated the approximated GAML that is introduced in Section 3.3. The results show that the approximated version can perform comparably to the original GAML. Though the approximated GAML shows slightly lower accuracies than the original GAML, it still performs superior to most of the existing algorithms.

4.3 Cross-domain few-shot classification

The cross-domain few-hot classification introduced by Chen et al. (2019) addresses a more challenging and practical few-shot classification scenario in which meta-train tasks and meta-test tasks are sampled from different task distributions. These scenarios are designed to evaluate meta-level overfitting of the meta-learning algorithms by creating a large domain gap between meta-trains and meta-tests. In particular, the algorithm can be said to be meta-overfitting if the algorithm relies too much on the prior knowledge of previously seen meta-train tasks instead of focusing on a few given examples to learn a new task. This meta-level overfitting makes the learning system more likely to fail to adapt to new tasks sampled from substantially different task distributions.

Datasets and experimental setup. To evaluate the level of meta-overfitting for GAML, we evaluate a cross-domain few-shot classification experiment. Mini-ImageNet is used for meta-train task, and Tiered-Imagenet (Ren et al., 2018), CUB-200-2011 (Wah et al., 2011), Cars (Bertinetto et al., 2018) are used for meta-test task. CUB has 200 fine-grained classes and consists of a total of 11,788 images. The CUB dataset is further divided into 100 meta-train classes, 50 meta-validation classes, and 50 meta-test classes. The Cars Krause et al. (2013) dataset consists of 16,185 images of 196 classes of cars. It is split into 8,144 training images and 8,041 testing images, where each class has been split roughly in 50-50. Classes are typically at the level of Make, Model, Year, e.g. 2012 Tesla Model S or 2012 BMW M3 coupe. As with the few-shot classification experiment, we use the standard *Conv-4* backbone and follow the same experimental protocol.

Results. Table 4 presents the cross-domain few-shot performance for GAML, MAML family, and PGD-MAML family. GAML significantly outperforms the state-of-the-art algorithms on 5-way 1-shot and 5-way 5-shot cross-domain classification tasks. In particular, for the Tiered-ImageNet, the performance was improved by 8.6% and 4.1% on 1-shot and 5-shot, respectively. Because GAML can simultaneously consider the task's individuality and optimization trajectory in the inner-loop optimization, it can overcome meta-overfitting better than the existing methods.

5 Discussion

Geometry-aware meta-learning: GAML employs a fully adaptive preconditioner such that the individual tasks with a wide diversity can have a chance of being handled well. As explained before,

GAML is the only PGD-MAML algorithm that corresponds to the full adaptation in Figure 1(e). A possible downside of the fully adaptive scheme, however, is that it can actually make the inner-loop optimization unreliable. To cope with this problem, it would make sense to constrain the preconditioner in a certain way such that desirable learning behaviors can be guaranteed. Riemannian metric is one of such possibilities where $\mathbf{P}$ is constrained to be a positive definite matrix. A choice of positive definite $\mathbf{P}$ corresponds to a choice of underlying structure, or geometry, in the parameter space (Amari, 1967; 1996; 1998). By meta-learning a positive definite $\mathbf{P}$, we are basically learning a knowledge on the geometry of the tasks such that the optimization path of the inner-loop can satisfy a desirable property.

If the choice of positive definite matrix $\mathbf{P}$ is a constant as shown in Figure 1(b-d), its geometric interpretation is also a constant in the parameter space. By pursuing a full adaptation as in Figure 1(e), we can implement a more sophisticated geometry-aware algorithm (Amari, 1996). Then, our goal is to design a preconditioner that is positive definite and also dependent on the task-specific parameter. In GAML, we have chosen diagonal matrices ($\phi = \{\mathbf{M}^1, \cdots, \mathbf{M}^l, \cdots, \mathbf{M}^L\}$) as the meta parameters and utilized SVD to allow $\mathbf{U}_{\tau,k}$ (left-side unitary matrix of the gradient matrix $\mathbf{G}_{\tau,k}$; see Eq. (12) in Supplementary B) to remain as a part of the preconditioning matrix. The dependency of $\mathbf{P}$ on $\mathbf{U}_{\tau,k}$, which is dependent on the task τ and the optimization-path, is the key for making GAML fully adaptive and geometry-aware. Our choice, however, is only one way of meeting the goal, and it remains as a limitation of our work to explore other possibilities.

Number of meta parameters: Recent MAML family and PGD-MAML family require a large increase in the number of meta-learning parameters as shown in Table 5. One advantage of GAML is that it requires only a very small increase in the number of meta parameters when compared to the baseline of MAML. This is made possible because we transform a gradient tensor into a gradient matrix, perform SVD of the matrix, and assign only a small number of meta parameters that correspond to the diagonal matrix of the gradient matrix. For the *Conv-4* network, GAML requires only 0.2% increase of the meta parameters. Even though the increase in the number of meta parameters is ignorable, still SVD of the gradient matrix can incur a large computational burden for very large networks. This is addressed by the approximated GAML.

Table 5: Comparison of the number of parameters for MAML, existing methods, and GAML.

Algorithm	# of params	% increase
MAML	1.2109×10^5	
Meta-SGD	2.4218×10^5	100.0%
MC	2.7106×10^6	2140.4%
PAMELA	1.6239×10^5	34.1%
MH	7.2196×10^7	59586.7%
Sparse-MAML	2.4218×10^5	100.0%
GAML	$\mathbf{1.2131 \times 10^5}$	**0.2%**

GAML vs. Approximated GAML: We can empirically show that the performance gap becomes smaller as the size of the network becomes larger. We trained *Conv-4* network with 32, 64, 128, and 256 channels and the performance gaps are shown in Figure 2. As expected, the performance gap becomes smaller as the network becomes larger. The gap is only around 0.8% when the channel size is 256. We are also showing the average cosine similarity between two randomly chosen row vectors of the preconditioned gradient $\tilde{\mathbf{G}}$'s, and the results are in line with Theorem 2 and Lemma 1 (Lemma 1 can be found in the Supplementary material).

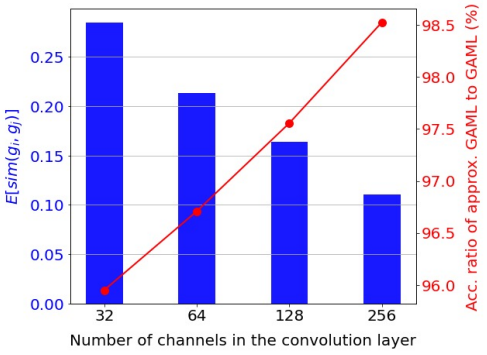


Figure 2: GAML vs. Approximated GAML

6 Conclusion

In this work, we have proposed GAML that is a PGD-MAML algorithm with a fully adaptive preconditioner. By utilizing a SVD operation, the preconditioner of GAML becomes a Riemannian metric as well. Thanks to the two desirable properties of being fully adaptive and being a Riemannian metric, GAML can handle individual tasks with a wide diversity within the MAML framework. GAML have achieved the state-of-the-art performances on a variety of few-shot learning tasks.

REFERENCES

Momin Abbas, Quan Xiao, Lisha Chen, Pin-Yu Chen, and Tianyi Chen. Sharp-maml: Sharpness-aware model-agnostic meta learning. *arXiv preprint arXiv:2206.03996*, 2022.

Alessandro Achille, Michael Lam, Rahul Tewari, Avinash Ravichandran, Subhransu Maji, Charless C Fowlkes, Stefano Soatto, and Pietro Perona. Task2vec: Task embedding for meta-learning. In *Proceedings of the IEEE/CVF international conference on computer vision*, pp. 6430–6439, 2019.

Shun-ichi Amari. Neural learning in structured parameter spaces-natural riemannian gradient. *Advances in neural information processing systems*, 9, 1996.

Shun-Ichi Amari. Natural gradient works efficiently in learning. *Neural computation*, 10(2):251–276, 1998.

Shun-Ichi Amari and Scott C Douglas. Why natural gradient? In *Proceedings of the 1998 IEEE International Conference on Acoustics, Speech and Signal Processing, ICASSP'98 (Cat. No. 98CH36181)*, volume 2, pp. 1213–1216. IEEE, 1998.

Shun-ichi Amari, Jimmy Ba, Roger Grosse, Xuechen Li, Atsushi Nitanda, Taiji Suzuki, Denny Wu, and Ji Xu. When does preconditioning help or hurt generalization? *arXiv preprint arXiv:2006.10732*, 2020.

Shunichi Amari. A theory of adaptive pattern classifiers. *IEEE Transactions on Electronic Computers*, (3):299–307, 1967.

Marcin Andrychowicz, Misha Denil, Sergio Gomez, Matthew W Hoffman, David Pfau, Tom Schaul, Brendan Shillingford, and Nando De Freitas. Learning to learn by gradient descent by gradient descent. *Advances in neural information processing systems*, 29, 2016.

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

Sungyong Baik, Myungsub Choi, Janghoon Choi, Heewon Kim, and Kyoung Mu Lee. Meta-learning with adaptive hyperparameters. *Advances in Neural Information Processing Systems*, 33:20755–20765, 2020a.

Sungyong Baik, Seokil Hong, and Kyoung Mu Lee. Learning to forget for meta-learning. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 2379–2387, 2020b.

Sungyong Baik, Janghoon Choi, Heewon Kim, Dohee Cho, Jaesik Min, and Kyoung Mu Lee. Meta-learning with task-adaptive loss function for few-shot learning. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pp. 9465–9474, 2021.

Luca Bertinetto, Joao F Henriques, Philip HS Torr, and Andrea Vedaldi. Meta-learning with differentiable closed-form solvers. *arXiv preprint arXiv:1805.08136*, 2018.

Irénée-Jules Bienaymé. *Considérations à l'appui de la découverte de Laplace sur la loi de probabilité dans la méthode des moindres carrés*. Imprimerie de Mallet-Bachelier, 1853.

Rishi Bommasani, Drew A Hudson, Ehsan Adeli, Russ Altman, Simran Arora, Sydney von Arx, Michael S Bernstein, Jeannette Bohg, Antoine Bosselut, Emma Brunskill, et al. On the opportunities and risks of foundation models. *arXiv preprint arXiv:2108.07258*, 2021.

Wei-Yu Chen, Yen-Cheng Liu, Zsolt Kira, Yu-Chiang Frank Wang, and Jia-Bin Huang. A closer look at few-shot classification. *arXiv preprint arXiv:1904.04232*, 2019.

Jia Deng, Wei Dong, Richard Socher, Li-Jia Li, Kai Li, and Li Fei-Fei. Imagenet: A large-scale hierarchical image database. In *2009 IEEE conference on computer vision and pattern recognition*, pp. 248–255. Ieee, 2009.

Lin Ding, Peng Liu, Wenfeng Shen, Weijia Lu, and Shengbo Chen. Gradient-based meta-learning using uncertainty to weigh loss for few-shot learning. *arXiv preprint arXiv:2208.08135*, 2022.

Carl Doersch, Ankush Gupta, and Andrew Zisserman. Crosstransformers: spatially-aware few-shot transfer. *Advances in Neural Information Processing Systems*, 33:21981–21993, 2020.

John Duchi, Elad Hazan, and Yoram Singer. Adaptive subgradient methods for online learning and stochastic optimization. *Journal of machine learning research*, 12(7), 2011.

Chen Fan, Parikshit Ram, and Sijia Liu. Sign-maml: Efficient model-agnostic meta-learning by signsgd. *arXiv preprint arXiv:2109.07497*, 2021.

Chelsea Finn, Pieter Abbeel, and Sergey Levine. Model-agnostic meta-learning for fast adaptation of deep networks. In *International conference on machine learning*, pp. 1126–1135. PMLR, 2017.

Chelsea Finn, Kelvin Xu, and Sergey Levine. Probabilistic model-agnostic meta-learning. *Advances in neural information processing systems*, 31, 2018.

Sebastian Flennerhag, Andrei A Rusu, Razvan Pascanu, Francesco Visin, Hujun Yin, and Raia Hadsell. Meta-learning with warped gradient descent. *arXiv preprint arXiv:1909.00025*, 2019.

Erin Grant, Chelsea Finn, Sergey Levine, Trevor Darrell, and Thomas Griffiths. Recasting gradient-based meta-learning as hierarchical bayes. *arXiv preprint arXiv:1801.08930*, 2018.

Markus Hiller, Mehrtash Harandi, and Tom Drummond. On enforcing better conditioned meta-learning for rapid few-shot adaptation. *arXiv preprint arXiv:2206.07260*, 2022.

David M Himmelblau et al. *Applied nonlinear programming*. McGraw-Hill, 2018.

Shell Xu Hu, Da Li, Jan Stühmer, Minyoung Kim, and Timothy M Hospedales. Pushing the limits of simple pipelines for few-shot learning: External data and fine-tuning make a difference. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 9068–9077, 2022.

Sergey Ioffe and Christian Szegedy. Batch normalization: Accelerating deep network training by reducing internal covariate shift. In *International conference on machine learning*, pp. 448–456. PMLR, 2015.

Sham M Kakade. A natural policy gradient. *Advances in neural information processing systems*, 14, 2001.

Diederik P Kingma and Jimmy Ba. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*, 2014.

Tamara G Kolda and Brett W Bader. Tensor decompositions and applications. *SIAM review*, 51(3): 455–500, 2009.

Jonathan Krause, Michael Stark, Jia Deng, and Li Fei-Fei. 3d object representations for fine-grained categorization. In *Proceedings of the IEEE international conference on computer vision workshops*, pp. 554–561, 2013.

Yann A LeCun, Léon Bottou, Genevieve B Orr, and Klaus-Robert Müller. Efficient backprop. In *Neural networks: Tricks of the trade*, pp. 9–48. Springer, 2012.

Sanghyuk Lee, Seunghyun Lee, and Byung Cheol Song. Contextual gradient scaling for few-shot learning. In *Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision*, pp. 834–843, 2022.

Yoonho Lee and Seungjin Choi. Gradient-based meta-learning with learned layerwise metric and subspace. In *International Conference on Machine Learning*, pp. 2927–2936. PMLR, 2018.

Chong Li and CJ Shi. Constrained optimization based low-rank approximation of deep neural networks. In *Proceedings of the European Conference on Computer Vision (ECCV)*, pp. 732–747, 2018.

Geng Li, Boyuan Ren, and Hongzhi Wang. Eeml: Ensemble embedded meta-learning. *arXiv preprint arXiv:2206.09195*, 2022.

Zhenguo Li, Fengwei Zhou, Fei Chen, and Hang Li. Meta-sgd: Learning to learn quickly for few-shot learning. *arXiv preprint arXiv:1707.09835*, 2017.

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.

Ahmed M Abdelmoniem, Ahmed Elzanaty, Mohamed-Slim Alouini, and Marco Canini. An efficient statistical-based gradient compression technique for distributed training systems. *Proceedings of Machine Learning and Systems*, 3:297–322, 2021.

Gabriel Maicas, Andrew P Bradley, Jacinto C Nascimento, Ian Reid, and Gustavo Carneiro. Training medical image analysis systems like radiologists. In *International Conference on Medical Image Computing and Computer-Assisted Intervention*, pp. 546–554. Springer, 2018.

Luckeciano C Melo. Transformers are meta-reinforcement learners. In *International Conference on Machine Learning*, pp. 15340–15359. PMLR, 2022.

Fei Mi, Minlie Huang, Jiyong Zhang, and Boi Faltings. Meta-learning for low-resource natural language generation in task-oriented dialogue systems. *arXiv preprint arXiv:1905.05644*, 2019.

Tsendsuren Munkhdalai and Hong Yu. Meta networks. In *International Conference on Machine Learning*, pp. 2554–2563. PMLR, 2017.

Tung Nguyen and Aditya Grover. Transformer neural processes: Uncertainty-aware meta learning via sequence modeling. *arXiv preprint arXiv:2207.04179*, 2022.

Alex Nichol and John Schulman. Reptile: a scalable metalearning algorithm. *arXiv preprint arXiv:1803.02999*, 2(3):4, 2018.

Jaehoon Oh, Hyungjun Yoo, ChangHwan Kim, and Se-Young Yun. Boil: Towards representation change for few-shot learning. *arXiv preprint arXiv:2008.08882*, 2020.

Eunbyung Park and Junier B Oliva. Meta-curvature. *Advances in Neural Information Processing Systems*, 32, 2019.

Danni Peng and Sinno Pan. Clustered task-aware meta-learning by learning from learning paths. 2021.

Juan-Manuel Perez-Rua, Xiatian Zhu, Timothy M Hospedales, and Tao Xiang. Incremental few-shot object detection. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 13846–13855, 2020.

M Przewięźlikowski, P Przybysz, J Tabor, M Zięba, and P Spurek. Hypermaml: Few-shot adaptation of deep models with hypernetworks. *arXiv preprint arXiv:2205.15745*, 2022.

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.

Jathushan Rajasegaran, Salman Khan, Munawar Hayat, Fahad Shahbaz Khan, and Mubarak Shah. Meta-learning the learning trends shared across tasks. *arXiv preprint arXiv:2010.09291*, 2020.

Aravind Rajeswaran, Chelsea Finn, Sham M Kakade, and Sergey Levine. Meta-learning with implicit gradients. *Advances in neural information processing systems*, 32, 2019.

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

Scott Reed, Konrad Zolna, Emilio Parisotto, Sergio Gomez Colmenarejo, Alexander Novikov, Gabriel Barth-Maron, Mai Gimenez, Yury Sulsky, Jackie Kay, Jost Tobias Springenberg, et al. A generalist agent. *arXiv preprint arXiv:2205.06175*, 2022.

Mengye Ren, Eleni Triantafillou, Sachin Ravi, Jake Snell, Kevin Swersky, Joshua B Tenenbaum, Hugo Larochelle, and Richard S Zemel. Meta-learning for semi-supervised few-shot classification. *arXiv preprint arXiv:1803.00676*, 2018.

Christian Simon, Piotr Koniusz, Richard Nock, and Mehrtash Harandi. On modulating the gradient for meta-learning. In *European Conference on Computer Vision*, pp. 556–572. Springer, 2020.

Rishav Singh, Vandana Bharti, Vishal Purohit, Abhinav Kumar, Amit Kumar Singh, and Sanjay Kumar Singh. Metamed: Few-shot medical image classification using gradient-based meta-learning. *Pattern Recognition*, 120:108111, 2021.

Jake Snell, Kevin Swersky, and Richard Zemel. Prototypical networks for few-shot learning. *Advances in neural information processing systems*, 30, 2017.

Xingyou Song, Yuxiang Yang, Krzysztof Choromanski, Ken Caluwaerts, Wenbo Gao, Chelsea Finn, and Jie Tan. Rapidly adaptable legged robots via evolutionary meta-learning. In *2020 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pp. 3769–3776. IEEE, 2020.

Oriol Vinyals, Charles Blundell, Timothy Lillicrap, Daan Wierstra, et al. Matching networks for one shot learning. *Advances in neural information processing systems*, 29, 2016.

Johannes Von Oswald, Dominic Zhao, Seijin Kobayashi, Simon Schug, Massimo Caccia, Nicolas Zucchet, and João Sacramento. Learning where to learn: Gradient sparsity in meta and continual learning. *Advances in Neural Information Processing Systems*, 34:5250–5263, 2021.

Catherine Wah, Steve Branson, Peter Welinder, Pietro Perona, and Serge Belongie. The caltech-ucsd birds-200-2011 dataset. 2011.

Shuhuan Wen, Zeteng Wen, Di Zhang, Hong Zhang, and Tao Wang. A multi-robot path-planning algorithm for autonomous navigation using meta-reinforcement learning based on transfer learning. *Applied Soft Computing*, 110:107605, 2021.

Wei Wen, Cong Xu, Chunpeng Wu, Yandan Wang, Yiran Chen, and Hai Li. Coordinating filters for faster deep neural networks. In *Proceedings of the IEEE International Conference on Computer Vision*, pp. 658–666, 2017.

Simon Wiedemann, Temesgen Mehari, Kevin Kepp, and Wojciech Samek. Dithered backprop: A sparse and quantized backpropagation algorithm for more efficient deep neural network training. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops*, pp. 720–721, 2020.

Xiongwei Wu, Doyen Sahoo, and Steven Hoi. Meta-rcnn: Meta learning for few-shot object detection. In *Proceedings of the 28th ACM International Conference on Multimedia*, pp. 1679–1687, 2020.

Yuhui Xu, Yuxi Li, Shuai Zhang, Wei Wen, Botao Wang, Wenrui Dai, Yingyong Qi, Yiran Chen, Weiyao Lin, and Hongkai Xiong. Trained rank pruning for efficient deep neural networks. In *2019 Fifth Workshop on Energy Efficient Machine Learning and Cognitive Computing-NeurIPS Edition (EMC2-NIPS)*, pp. 14–17. IEEE, 2019.

Peng Yang, Shaogang Ren, Yang Zhao, and Ping Li. Calibrating cnns for few-shot meta learning. In *Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision*, pp. 2090–2099, 2022.

Huaxiu Yao, Xian Wu, Zhiqiang Tao, Yaliang Li, Bolin Ding, Ruirui Li, and Zhenhui Li. Automated relational meta-learning. *arXiv preprint arXiv:2001.00745*, 2020.

Jaesik Yoon, Taesup Kim, Ousmane Dia, Sungwoong Kim, Yoshua Bengio, and Sungjin Ahn. Bayesian model-agnostic meta-learning. *Advances in neural information processing systems*, 31, 2018.

Dominic Zhao, Johannes von Oswald, Seijin Kobayashi, João Sacramento, and Benjamin F Grewe. Meta-learning via hypernetworks. 2020.

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

Supplementary materials for the paper "GAML: geometry-aware meta-learning via a fully adaptive preconditioner"

A HYPER-PARAMETERS SETTING

In this section, we provide the details of hyper-parameters of GAML on various few-shot learning tasks.

Hyper-parameter	Sinusoid			Mini-imagenet		Tiered-imagenet		Cross-domain	
	5 shot	10 shot	20 shot	1 shot	5 shot	1 shot	5 shot	1 shot	5 shot
Bathc size	4	4	4	4	2	4	2	4	4
Total training iteration	70000	70000	70000	60000	60000	130000	200000	60000	60000
inner learning rate α	0.01	0.01	0.01	0.01	0.01	0.01	0.01	0.01	0.01
outer learning rate β_1	0.001	0.001	0.001	0.0001	0.0001	0.0001	0.0001	0.0001	0.0001
outer learning rate β_2	0.001	0.001	0.0001	0.003	0.0005	0.003	0.0001	0.003	0.0005
The number of training inner steps	5	5	5	5	5	5	5	5	5
The number of testing inner steps	10	10	10	10	10	10	10	10	10
Data augmentation	None			random flip		random flip		random flip	

Table A.1: Hyper-parameters used for training GAML with standard 4-Conv and 2-layer MLP on various few-shot learning experiments.

B PROOFS

This section presents the proofs of the theorem and lemma stated in Section 3.

Theorem 1. (Restated) *Let $\tilde{\mathbf{G}}^l_{\tau,k} \in \mathbb{R}^{m\times n}$ be the 'l-layer k-th inner-step gradient' transformed by meta parameter $\mathbf{M}^l$ for task τ. Then preconditioner $\mathbf{P}_{GAML}$ induced by $\tilde{\mathbf{G}}^l_{\tau,k}$ is a Riemannian metric and depends on the task-specific parameter $\theta_{\tau,k}$.*

Proof. We can rewrite the $\tilde{\mathbf{G}}^l_{\tau,k}$ as follows:

$$\tilde{\mathbf{G}}^l_{\tau,k} = \mathbf{U}^l_{\tau,k}(\mathbf{M}^l \cdot \mathbf{\Sigma}^l_{\tau,k}){\mathbf{V}^l_{\tau,k}}^T \tag{12}$$

$$= (\mathbf{U}^l_{\tau,k}\mathbf{M}^l{\mathbf{U}^l_{\tau,k}}^T)\mathbf{U}^l_{\tau,k}\mathbf{\Sigma}^l_{\tau,k}{\mathbf{V}^l_{\tau,k}}^T \tag{13}$$

$$= \mathbf{D}^l_{\tau,k}\mathbf{G}^l_{\tau,k}, \tag{14}$$

where $\mathbf{D}^l_{\tau,k} = \mathbf{U}^l_{\tau,k}\mathbf{M}^l{\mathbf{U}^l_{\tau,k}}^T$. To induce preconditioner in the Eq. (12), we transform Eq. (12) as the general gradient descent form (matrix-vector product):

$$\mathrm{vec}(\tilde{\mathbf{G}}^l_{\tau,k}) = \mathrm{blkdiag}(\underbrace{\mathbf{D}^l_{\tau,k}, \cdots, \mathbf{D}^l_{\tau,k}}_{n \text{ times}}) \cdot \mathrm{vec}(\mathbf{G}^l_{\tau,k}) \tag{15}$$

$$= \mathbf{P}_{\mathrm{GAML}} \cdot \mathrm{vec}(\mathbf{G}^l_{\tau,k}), \tag{16}$$

where $\mathbf{P}_{\mathrm{GAML}}$ is a block diagonal matrix such that the main-diagonal blocks are $\mathbf{D}^l_{\tau,k}$'s. Since $\mathbf{D}^l_{\tau,k}$ is trivially symmetric positive definite, $\mathbf{P}_{\mathrm{GAML}}$ is symmetric positive definite. Therefore $\mathbf{P}_{\mathrm{GAML}}$ is a Riemannian metric. Since the unitary matrix $\mathbf{U}^l_{\tau,k}$ depends on the gradient $\tilde{\mathbf{G}}^l_{\tau,k}$, it trivially depends on the task-wise parameters $\theta_{\tau,k}$. Therefore $\mathbf{P}_{\mathrm{GAML}}$ depends on the task-wise parameters $\theta_{\tau,k}$ since it depends on the unitary matrix $\mathbf{U}^l_{\tau,k}$. □

Lemma 1. *If a random vectors $\boldsymbol{x} = (X_1, \cdots, X_n) \in \mathbb{R}^n$ has an uniform distribution on the $(n-1)$-dimensional unit sphere, then*

$$\mathbb{V}(X_i) = \frac{1}{n}. \tag{17}$$

Proof. Since $X_1, \cdots, X_n$ follow an identical distribution, $\mathbb{V}(X_i) = \mathbb{V}(X_j)$ holds for all i, j. Thus,

$$n\mathbb{V}(X_i) = \sum_{i=1}^{n} \mathbb{V}(X_i). \tag{18}$$

Then, we compute the sum of variance as follows:

$$\sum_{i=1}^{n} \mathbb{V}(X_i) = \sum_{i=1}^{n} \mathbb{E}(X_i^2) \text{ (by } \mathbb{E}(X) = 0) \tag{19}$$

$$= \mathbb{E}(\sum_{i=1}^{n} X_i^2) \tag{20}$$

$$= \mathbb{E}(||X||_2^2) \tag{21}$$

$$= 1. \tag{22}$$

Using Equation 18 and 19, we have

$$\mathbb{V}(X_i) = \frac{1}{n}. \tag{23}$$

□

Lemma 2. *If two independent random vectors $\boldsymbol{x} = (X_1, \cdots, X_n)$, $\boldsymbol{y} = (Y_1, \cdots, Y_n) \in \mathbb{R}^n$ follow a uniform distribution on the $(n-1)$-dimensional unit sphere, then*

$$P(|\langle \boldsymbol{x}, \boldsymbol{y} \rangle| > \epsilon) \leq \frac{1}{n\epsilon^2}. \tag{24}$$

Proof. Since we can rotate coordinate so that $\boldsymbol{y} = (1, 0, \cdots, 0) \in \mathbb{R}^n$, we have

$$\langle \boldsymbol{x}, \boldsymbol{y} \rangle = X_1. \tag{25}$$

Following Equation 25, we show that its expectation is equal to:

$$\mathbb{E}[\langle \boldsymbol{x}, \boldsymbol{y} \rangle] = \mathbb{E}[X_1], \tag{26}$$

$$= 0 \tag{27}$$

and its variance is equal to:

$$\mathbb{V}[\langle \boldsymbol{x}, \boldsymbol{y} \rangle] = \mathbb{V}[X_1], \tag{28}$$

$$= \frac{1}{n} \text{ (by Lemma 1).} \tag{29}$$

By applying Chebyshev's inequality (Bienaymé, 1853) on $\langle \boldsymbol{x}, \boldsymbol{y} \rangle$, we have

$$P(|\langle \boldsymbol{x}, \boldsymbol{y} \rangle| \geq \frac{k}{\sqrt{n}}) \leq \frac{1}{k^2}, \tag{30}$$

for any real number $k > 0$. Let $\frac{k}{\sqrt{n}}$ be a ϵ. Then we rewrite the inEq. (30) as follows:

$$P(|\langle \boldsymbol{x}, \boldsymbol{y} \rangle| \geq \epsilon) \leq \frac{1}{n\epsilon^2}. \tag{31}$$

This result indicates that the two vectors $\boldsymbol{x}$ and $\boldsymbol{y}$ become asymptotically orthogonal as n increases.
□

Theorem 2. (Restated) *Let $\mathbf{G} \in \mathbb{R}^{m \times n}$ be a gradient tensor reshaped into a matrix form. As n becomes large, GAML's preconditioned gradient matrix $\tilde{\mathbf{G}}$ asymptotically becomes equivalent to $\mathbf{MG}$, where $\mathbf{M}$ is a diagonal matrix. Therefore, the following holds for a large n.*

$$\tilde{\mathbf{G}} \cong \mathbf{MG}. \tag{32}$$

Proof. Let $\boldsymbol{g}_1, \boldsymbol{g}_2, \cdots, \boldsymbol{g}_m$ are the row vectors of $\mathbf{G}$. Then,

$$\mathbf{G} = \begin{bmatrix} \|\boldsymbol{g}_1\| & & \\ & \ddots & \\ & & \|\boldsymbol{g}_m\| \end{bmatrix} \begin{bmatrix} \frac{\boldsymbol{g}_1}{\|\boldsymbol{g}_1\|} \\ \vdots \\ \frac{\boldsymbol{g}_m}{\|\boldsymbol{g}_m\|} \end{bmatrix}. \tag{33}$$

In (Wiedemann et al., 2020; M Abdelmoniem et al., 2021), some prior works have taken the assumption that the gradient tensor is sampled from an i.i.d. normal distribution. Following them, we assume that the elements of $\mathbf{G}$ have an i.i.d. normal distribution. With the assumption, we have

$$\frac{\boldsymbol{g}_i}{\|\boldsymbol{g}_i\|} \perp\!\!\!\perp \frac{\boldsymbol{g}_j}{\|\boldsymbol{g}_j\|} \ (\forall i \neq j). \tag{34}$$

Since independent vectors $\frac{\boldsymbol{g}_i}{\|\boldsymbol{g}_i\|}, \frac{\boldsymbol{g}_j}{\|\boldsymbol{g}_j\|}$ are located on the $(n-1)$-dimensional unit sphere, the vectors are asymptotically orthogonal as n increases by Lemma 1.

Now that we rewrite $\mathbf{G}$ as follows:

$$\mathbf{G} = \mathbf{I} \begin{bmatrix} \|\boldsymbol{g}_1\| & & \\ & \ddots & \\ & & \|\boldsymbol{g}_m\| \end{bmatrix} \begin{bmatrix} \frac{\boldsymbol{g}_1}{\|\boldsymbol{g}_1\|} \\ \vdots \\ \frac{\boldsymbol{g}_m}{\|\boldsymbol{g}_m\|} \end{bmatrix}. \tag{35}$$

Since $\mathbf{I}$ is the unitary martrix and $(\frac{\boldsymbol{g}_1}{\|\boldsymbol{g}_1\|}, \cdots, \frac{\boldsymbol{g}_m}{\|\boldsymbol{g}_m\|})^T$ is approximated to semi-unitary matrices with large n, the singular values of $\mathbf{G}$ are approximated to $\|\boldsymbol{g}_1\|, \cdots, \|\boldsymbol{g}_m\|$.

In order to transform the singular values of $\mathbf{G}$ where n is large enough, we can simply multiply $\mathbf{M}$ to $\mathbf{G}$. Therefore,

$$\tilde{\mathbf{G}} \cong \mathbf{M}\mathbf{G}. \tag{36}$$

□

C ADDITIONAL COMPARISON RESULTS

More researches related to GAML are compared and summarized in Table C.2

Table C.2: 5-way few-shot classification accuracy (%) on mini-ImageNet with a *Conv-4* backbone. We report mean ± 95% confidence intervals(ci) for 600 tasks according to Finn et al. (2017). † denotes PGD-MAML family.

Algorithm	5-way 1-shot	5-way 5-shot
MAML (Finn et al., 2017)	47.89 ± 1.20	64.59 ± 0.88
Meta-SGD† (Li et al., 2017)	50.47 ± 1.87	64.00 ± 0.90
BMAML (Yoon et al., 2018)	53.80 ± 1.46	64.23 ± 0.69
ANIL (Raghu et al., 2019)	46.70 ± 0.40	61.50 ± 0.50
LLAMA (Grant et al., 2018).	49.40 ± 1.83	N/A
PLATIPUS (Finn et al., 2018)	50.13 ± 1.86	-
T-net† (Lee & Choi, 2018)	50.86 ± 1.82	N/A
MT-net† (Lee & Choi, 2018)	51.70 ± 1.84	N/A
MAML++ (Antoniou et al., 2018)	52.15 ± 0.26	68.32 ± 0.44
iMAML-HF (Rajeswaran et al., 2019)	49.30 ± 1.88	N/A
WarpGrad (Flennerhag et al., 2019)	52.30 ± 0.90	68.40 ± 0.60
MC1† (Park & Oliva, 2019)	53.74 ± 0.84	68.01 ± 0.73
MC2† (Park & Oliva, 2019)	54.08 ± 0.88	67.99 ± 0.73
MH-C† (Zhao et al., 2020)	48.64 ± 0.33	64.52 ± 0.51
MH† (Zhao et al., 2020)	49.41 ± 0.96	67.16 ± 0.42
BOIL (Oh et al., 2020)	49.61 ± 0.16	66.46 ± 0.37
ARML (Yao et al., 2020)	50.42 ± 1.79	64.12 ± 0.90
ALFA (Baik et al., 2020a)	50.58 ± 0.51	69.12 ± 0.47
L2F (Baik et al., 2020b)	52.10 ± 0.50	69.38 ± 0.46
ModGrad† (Simon et al., 2020)	53.20 ± 0.86	69.17 ± 0.69
PAMELA† (Rajasegaran et al., 2020)	53.50 ± 0.89	70.51 ± 0.67
SignMAML (Fan et al., 2021)	42.90 ± 1.50	60.70 ± 0.70
CTML (Peng & Pan, 2021)	50.47 ± 1.83	64.15 ± 0.90
MeTAL (Baik et al., 2021)	52.63 ± 0.37	70.52 ± 0.29
ECML (Hiller et al., 2022)	48.94 ± 0.80	65.26 ± 0.67
Sharp-MAML_up (Abbas et al., 2022)	$49.56 \pm$ N/A	$63.06 \pm$ N/A
Sharp-MAML_low (Abbas et al., 2022)	$49.72 \pm$ N/A	$63.18 \pm$ N/A
Sharp-MAML_both (Abbas et al., 2022)	$50.28 \pm$ N/A	$65.04 \pm$ N/A
FBM (Yang et al., 2022)	50.62 ± 1.79	64.78 ± 0.35
CxGrad (Lee et al., 2022)	51.80 ± 0.46	69.82 ± 0.42
HyperMAML (Przewięźlikowski et al., 2022)	51.84 ± 0.57	66.29 ± 0.43
EEML (Li et al., 2022)	52.42 ± 1.75	68.40 ± 0.95
MH-O† (Zhao et al., 2020)	52.50 ± 0.61	67.76 ± 0.34
Sparse-MAML† (Von Oswald et al., 2021)	50.35 ± 0.39	67.03 ± 0.74
Sparse-ReLU-MAML† (Von Oswald et al., 2021)	50.39 ± 0.89	64.84 ± 0.46
Sparse-MAML+† (Von Oswald et al., 2021)	51.04 ± 0.59	67.03 ± 0.74
GAML (approx.)†	53.52 ± 0.88	70.75 ± 0.67
GAML†	$\mathbf{54.86 \pm 0.85}$	$\mathbf{71.55 \pm 0.61}$