On Modulating the Gradient for Meta-Learning

Christian Simon[†,§] Piotr Koniusz[†,§]
Richard Nock[†,‡,§] Mehrtash Harandi[♣,§]

[†]The Australian National University, [♣]Monash University,
[‡]The University of Sydney, [§] Data61-CSIRO
first.last@{anu.edu.au,monash.edu,data61.csiro.au}

Abstract. Inspired by optimization techniques, we propose a novel meta-learning algorithm with gradient modulation to encourage fast-adaptation of neural networks in the absence of abundant data. Our method, termed ModGrad, is designed to circumvent the noisy nature of the gradients which is prevalent in low-data regimes. Furthermore and having the scalability concern in mind, we formulate ModGrad via low-rank approximations, which in turn enables us to employ ModGrad to adapt hefty neural networks. We thoroughly assess and contrast ModGrad against a large family of meta-learning techniques and observe that the proposed algorithm outperforms baselines comfortably while enjoying faster convergence.

Keywords: Meta Learning, Few Shot Learning, Adaptive Gradients

1 Introduction

Gradient-based algorithms [3, 9, 31–33, 45] can be successfully employed to address a broad set of problems in meta-learning including image classification [4, 37], regression [9, 45], and reinforcement learning [1] to name a few. Despite the success, employing gradient-based methods in the absence of abundant data (*e.g.*, few-shot learning) is daunting as gradients become *noisy* [44]. To outline the setup, consider training a neural network with parameters $\boldsymbol{\theta} \in \Theta$. Let

$$\mathcal{L}_N(\boldsymbol{\theta}) = \frac{1}{N}\sum_{i=1}^{N} \ell(\boldsymbol{x}_i, y_i, \boldsymbol{\theta}),$$

be the total loss of the network where $\ell : \mathcal{X} \times \mathcal{Y} \times \Theta \to \mathbb{R}^+$ denotes the loss for the input $\boldsymbol{x}_i \in \mathcal{X}$ and desired output $y_i \in \mathcal{Y}$. In majority of cases, training proceeds by computing the gradient $\boldsymbol{g} = \nabla_{\boldsymbol{\theta}}\mathcal{L}_N(\boldsymbol{\theta}) = \frac{\partial}{\partial \boldsymbol{\theta}}\mathcal{L}_N(\boldsymbol{\theta})$ followed by updates in the form

$$\boldsymbol{\theta}^{(k)} = \boldsymbol{\theta}^{(k-1)} - \alpha \nabla_{\boldsymbol{\theta}^{k-1}} \mathcal{L}_N\left(\boldsymbol{\theta}^{(k-1)}\right) . \quad (1)$$

Upon availability of ample data (a big enough N), updates using the gradient will hopefully converge to a good minimum. The dependency of the gradient on N (number of samples) immediately suggests the noisy nature of $\boldsymbol{g}$ in the low-data regime. That is, if we can only see $n \ll N$ samples,

$$\hat{\boldsymbol{g}} = \nabla_{\boldsymbol{\theta}} \mathcal{L}_n(\boldsymbol{\theta}) = \frac{\partial}{\partial \boldsymbol{\theta}} \frac{1}{n} \sum_{i=1}^{n} \ell(\boldsymbol{x}_i, y_i, \boldsymbol{\theta}) \approx \boldsymbol{g} \ .$$

The issue intensifies if the learning algorithm benefits from the Hessian information or if the distribution of data changes. In meta-learning, the use of former might be tempting to achieve faster convergence while the latter occurs frequently due to the nature of the problem (*e.g.*, adaptation to new tasks).

To combat this issue, one would ideally like to use smaller learning rates for the noisy elements of the gradient, hence reducing their effects. This is indeed the underlying idea of some modern meta-learning algorithms, one way or another.

For example, the Meta-SGD [22] algorithm explicitly formulates the learning problem as finding meta-parameters of the network along their optimal learning rate. In LEO [33], updates are performed in a low-dimensional space and the result is consequently projected to the parameter space using a non-linear mapping.

A tool from optimization, called preconditioning, is a principled way for accelerating the convergence rate of the first-order methods. The idea is that instead of updates in the form of Eq. (1), one would use

$$\boldsymbol{\theta}^{(k)} = \boldsymbol{\theta}^{(k-1)} - \alpha \boldsymbol{P} \boldsymbol{g}\big(\boldsymbol{\theta}^{(k-1)}\big) \ . \tag{2}$$

Obviously, if the preconditioner is chosen to be the inverse of the Hessian matrix (*ie.*, $\boldsymbol{P} = \boldsymbol{H}^{-1}$ for $\boldsymbol{H} = \frac{\partial^2}{\partial \boldsymbol{\theta} \partial \boldsymbol{\theta}^\top}$), or approximates $\boldsymbol{H}^{-1}$ well enough for that matter, preconditioning reduces to the Newton method which enjoys a quadratic convergence rate. Inspecting Eq. 2 raises a question if it *is possible to use the idea of preconditioning to address the noisy gradient problem in meta-learning?* To answer this question, we hypothesize that in contrast to a full preconditioning matrix, using just the diagonal preconditioner may effectively alter the learning rate in an element-wise fashion and provide acceleration of the convergence of the learner. In this paper, we study this particular idea in detail. In particular and inspired by the concept of preconditioning, we make the following contributions.

Contributions.

1. We propose a meta-learning algorithm to learn to modulate the gradient in the absence of abundant data. Similarly to preconditioning and as the name implies, the gradient modulation is a multiplicative corrective factor albeit task-dependent. Being vigilant about the scalability, we formulate the modulation via low-rank approximation, which in turn let us apply modulation on large models. Such an idea is significantly different from previous works which suffer from poor scalability and require adaptation of parts of the network (usually the classifier).

2. We extensively compare and contrast our algorithm against state-of-the-art methods on several tasks, ranging from image classification to image completion

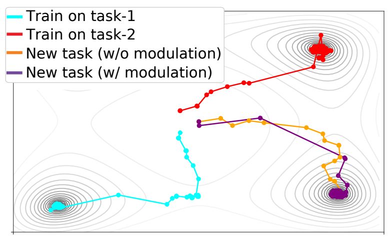


Fig. 1: An illustration of our modulation applied for a new task after learning from two previous tasks (task-1 and task-2).

to reinforcement learning. Empirically, our method outperforms various state-of-the-art algorithms and exhibits faster convergence (see Fig. 1 for an illustration).
3. We further study the robustness of the meta-learning algorithms in the presence of corrupted gradients. We observe that our idea of modulating the gradient is able to recover gracefully while other methods severely underperform.

2 Background

The objective of meta-learning is to **1.** achieve rapid convergence for new tasks (task-level) and **2.** generalize beyond previously seen tasks (meta-level). A common approach to meta-learning is to design models that learn from limited data using the concept of episodic training [34,39]. Therein, a model is presented with a set of tasks (*e.g.*, image classification), where for each task, only limited data is available. To put the discussion into context, we first provide a brief overview of the Model Agnostic Meta Learning (MAML) algorithm [9]. Let $\mathcal{D}_\tau^{trn}$ and $\mathcal{D}_\tau^{val}$ be the training and the validation sets of a given task $\tau \sim p(\mathcal{T})$, respectively. We assume that $\mathcal{D} \coloneqq \{\boldsymbol{x}_i, y_i\}_{i=1}^{|\mathcal{D}|}, \boldsymbol{x}_i \in \mathcal{X},\ y_i \in \mathcal{Y}$ for some small $|\mathcal{D}|$. Furthermore, let $h : \mathcal{X} \times \mathbb{R}^n \to \mathcal{Y}$ be the predictor function of the model parameterized by $\boldsymbol{\theta} \in \mathbb{R}^n$. The MAML algorithm seeks a universal initialization $\boldsymbol{\theta}^*$ by minimizing:

$$\min_{\boldsymbol{\theta}^*} \sum_{\tau \sim P(\mathcal{T})} \mathcal{L}\Big(D_\tau^{\text{val}}, \boldsymbol{\theta}^* - \alpha \sum_{k=0}^{(K-1)} \nabla \mathcal{L}\big(D_\tau^{\text{trn}}, \boldsymbol{\theta}_\tau^{(k)}\big)\Big). \tag{3}$$

Here, $\boldsymbol{\theta}_\tau^{(k)} = \boldsymbol{\theta}_\tau^{(k-1)} - \alpha \nabla \mathcal{L}\big(D_\tau^{\text{trn}}, \boldsymbol{\theta}_\tau^{(k-1)}\big)$ with $\boldsymbol{\theta}_\tau^0 = \boldsymbol{\theta}^*$. The loss terms are:

$$\begin{aligned} \mathcal{L}\big(D_\tau^{\text{trn}}, \boldsymbol{\theta}\big) &\coloneqq \mathbb{E}_{\boldsymbol{x},y \sim \mathcal{D}_\tau^{\text{trn}}} \big[\ell(h(\boldsymbol{x}, \boldsymbol{\theta}), y)\big], \\ \mathcal{L}\big(D_\tau^{\text{val}}, \boldsymbol{\theta}\big) &\coloneqq \mathbb{E}_{\boldsymbol{x},y \sim \mathcal{D}_\tau^{\text{val}}} \big[\ell(h(\boldsymbol{x}, \boldsymbol{\theta}), y)\big]. \end{aligned} \tag{4}$$

Intuitively, given a task τ, the MAML starts from $\boldsymbol{\theta}^*$ and performs K gradient updates on D_τ^{trn} to obtain the adapted parameters $\boldsymbol{\theta}_\tau^{(K)}$ (this is called the inner-loop updates). Then it uses D_τ^{val} and $\boldsymbol{\theta}_\tau^{(K)}$ (which is dependent on $\boldsymbol{\theta}^*$) to improve

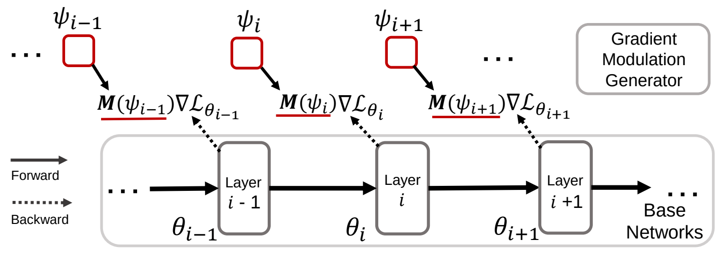

ψ_{i-1}
ψ_i
ψ_{i+1}
Gradient Modulation Generator
$M(\psi_{i-1})\nabla\mathcal{L}_{\theta_{i-1}}$
$M(\psi_i)\nabla\mathcal{L}_{\theta_i}$
$M(\psi_{i+1})\nabla\mathcal{L}_{\theta_{i+1}}$
Forward
Backward
Layer i - 1
Layer i
Layer i +1
θ_{i-1}
θ_i
θ_{i+1}
Base Networks

Fig. 2: Every layer is equipped with a gradient correction generator to modulate the incoming gradients.

the universal initialization point $\boldsymbol{\theta}^*$ (this is called the outer-loop update). The extensions of MAML [9] include Meta-SGD [22] and Meta-Curvature [27] with adaptive learning rates, CAVIA [45] with feature modulation, and LEO [33] with low dimensional updates.

3 Proposed Method

In this section, we introduce our meta-learner to accelerate learning process for few- and multi-shot classification, regression, and reinforcement learning. Our proposed method essentially learns to **Mod**ulate the **Grad**ient via so-called meta-learning **ModGrad** such that each task has a specific gradient modulator depending on the context.

3.1 Inner-Loop with Gradient Modulation

We define $\boldsymbol{M} : \mathbb{R}^d \to \mathbb{R}^n$, a function with parameter $\boldsymbol{\Psi}$ that performs gradient modulation. The purpose of the gradient modulation is two-fold: **1.** to suppress the noise and **2.** accelerate the convergence by amplifying certain elements of the gradient. Inspired by the use of diagonal for preconditioning, we formulate the meta-learning algorithm as follows:

$$\min_{\boldsymbol{\theta}^*,\boldsymbol{\Psi}} \sum_{\tau\sim P(\mathcal{T})} \mathcal{L}\Big(D_\tau^{\text{val}}, \boldsymbol{\theta}^* - \alpha \sum_{k=0}^{K-1} \boldsymbol{M}_\tau^{(k)}(\boldsymbol{\Psi}) \odot \nabla\mathcal{L}\big(D_\tau^{\text{trn}}, \boldsymbol{\theta}_\tau^{(k)}\big)\Big). \tag{5}$$

The inner-loop update in Eq. 5 performs an element-wise modulation of the gradient vector by operator '$\odot$'. One can also view this updating scheme as a generalization of meta-learners that adaptively alter the learning rate of the SGD (*e.g.*, [3,22,33]). In what follows next, we build a generative modulator through another neural network.

3.2 Task-Dependent Gradient Modulation Generator

By minimizing Eq. 5, we jointly learn the universal initialization vector $\boldsymbol{\theta}^*$ and the generator $\boldsymbol{M}_\tau^{(k)}(\boldsymbol{\Psi})$ to enrich adaptability of the meta-learner. Fig. 2 and 3

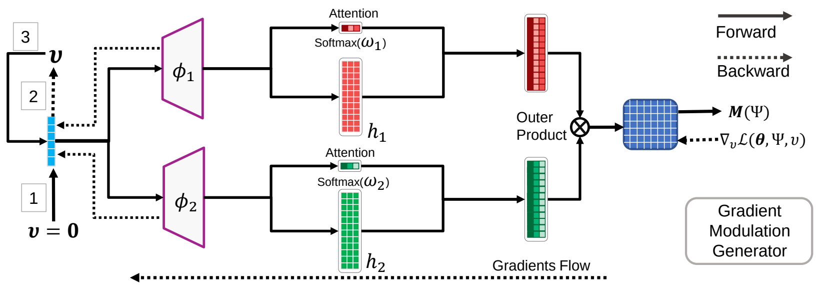


Fig. 3: A gradient modulation generator. Two sister networks (ϕ_1, ϕ_2) produce two tall matrices to generate the correction.

illustrate our design to generate the modulation. Without loss of generality, we denote $n = n_1 \times n_2$ and we elaborate on how the function $\boldsymbol{M}_\tau^{(k)}(\boldsymbol{\Psi}) \in \mathbb{R}^n$ is obtained via the generator for a given task τ.

For reasons that become clear shortly, the generator makes use of a context vector $\boldsymbol{\nu} \in \mathbb{R}^d$ to generate the gradient modulation $\boldsymbol{M}_\tau^{(k)}(\boldsymbol{\Psi})$. In doing so, the context vector $\boldsymbol{\nu}$ is first processed by two sister modules ϕ_1 and ϕ_2. This generates,

$$\begin{aligned} (\boldsymbol{\omega}_1, \boldsymbol{h}_1) &= \phi_1(\boldsymbol{\nu}_\tau^{(k)}), \quad \boldsymbol{\omega}_1 \in \mathbb{R}^u, \boldsymbol{h}_1 \in \mathbb{R}^{n_1 \times u}, \\ (\boldsymbol{\omega}_2, \boldsymbol{h}_2) &= \phi_2(\boldsymbol{\nu}_\tau^{(k)}), \quad \boldsymbol{\omega}_2 \in \mathbb{R}^u, \boldsymbol{h}_2 \in \mathbb{R}^{n_2 \times u}. \end{aligned} \tag{6}$$

Attention mechanism is employed to re-weigh the matrix produced by two sister modules:

$$\begin{aligned} \boldsymbol{a}_1 &= \mathrm{softmax}(\boldsymbol{\omega}_1), \\ \boldsymbol{a}_2 &= \mathrm{softmax}(\boldsymbol{\omega}_2). \end{aligned} \tag{7}$$

The softmax function is chosen because we observe empirically that it always performs the best compared to other activation functions. The output of the gradient modulator is then computed by the outer product operation:

$$\boldsymbol{M}_\tau^{(k)}(\boldsymbol{\Psi}) = \mathrm{Vec}\big((\boldsymbol{h}_1\boldsymbol{a}_1) \otimes (\boldsymbol{h}_2\boldsymbol{a}_2)\big). \tag{8}$$

Vec($\cdot$) operator vectorizes an input matrix. Eq. 8 uses a low-rank approximation to generate gradient corrections. This lets us scale up ModGrad to very large networks and simultaneously regularizes gradients. Finally, the produced matrix is passed via ReLU.

Finally, the only remaining detail of the ModGrad algorithm is the context vector generation. To this end, we firstly reset $\boldsymbol{\nu}_\tau^{(0)} = \mathbf{0}$ and then we generate an initial modulation $\boldsymbol{M}_\tau^{(k)}(\boldsymbol{\Psi})$. We then update $\boldsymbol{\nu}$ as:

$$\boldsymbol{\nu}_\tau^{(k)} = -\nabla_{\boldsymbol{\nu}_\tau^{(k-1)}} \mathcal{L}\big(\boldsymbol{\theta}^{(k-1)} - \alpha \boldsymbol{M}_\tau^{(k-1)}(\boldsymbol{\Psi}) \odot \nabla_{\boldsymbol{\theta}^{(k-1)}} \mathcal{L}(\boldsymbol{\theta}^{(k-1)})\big) \,. \tag{9}$$

The context vector $\boldsymbol{\nu}$ is then fed into two sister networks (Eq. 6) to compute the modulation. Algorithm 1, outlined for classification and regression tasks, provides details of how the parameters of the generator $\boldsymbol{\Psi}$ and the base network $\boldsymbol{\theta}$ are updated. For the reinforcement learning, another variant of ModGrad is provided in the supplementary material.

Algorithm 1 Train ModGrad

```
1:  **Require**: $\boldsymbol{\theta}$, $\boldsymbol{\Psi}$, $\alpha$, $p(\mathcal{T})$
2:  $\boldsymbol{\theta}$, $\boldsymbol{\Psi} \leftarrow$ Random initialization
3:  **while** not done **do**
4:      Sample $\tau_1 \dots \tau_B$ from $p(\mathcal{T})$                      ▷ Sample episodes
5:      **for** $b$ in $\{1, ..., B\}$ **do**
6:          $\boldsymbol{\theta}^0 \leftarrow \boldsymbol{\theta}$
7:          $\mathcal{D}_\tau^{trn}, \mathcal{D}_\tau^{val}$ from $\tau_b$              ▷ Sample training and testing sets
8:          **for** $k$ in $\{1, ..., K\}$ **do**
9:              Reset $\boldsymbol{\nu}$                                  ▷ Reset context vector to 0
10:             Compute $\nabla_{\boldsymbol{\theta}^{(k-1)}} \mathcal{L}(\boldsymbol{\theta}^{(k-1)})$
11:             Compute $\boldsymbol{\nu}$ using Eq. 9                      ▷ Update context vector
12:             Generate the gradient modulation using Eq. 8
13:             $\boldsymbol{\theta}^{(k)} \leftarrow \boldsymbol{\theta}^{(k-1)} - \alpha \boldsymbol{M}_\tau^{(k-1)}(\boldsymbol{\Psi}) \odot \nabla_{\boldsymbol{\theta}^{(k-1)}} \mathcal{L}(\boldsymbol{\theta}^{(k-1)})$
14:         **end for**
15:     **end for**
16:     $\boldsymbol{\theta} \leftarrow$ OptimizerStep($\mathcal{D}_\tau^{val}, \boldsymbol{\theta}^{(K)}$)          ▷ Meta update base networks
17:     $\boldsymbol{\Psi} \leftarrow$ OptimizerStep($\mathcal{D}_\tau^{val}, \boldsymbol{\Psi}$)             ▷ Meta update ModGrad
18: **end while**
```

Remark 1. ModGrad uses a lower-dimensional context vector $\boldsymbol{\nu}$ to generate the gradient modulation. This enables us to lower the computational complexity even further. The underlying assumption, based on the smoothness of the gradient updates, is that the gradient field, especially for modern models which are often deep and over-parameterized, should comply with the low-dimensional latent data representation. As such, enforcing the context vector to be low-dimensional implicitly contributes to capturing the geometry of the gradient field.

Remark 2. In contrast to T-Net [21] and natural neural networks [8] that alter the neural networks by inserting additional layers for gradient projection, ModGrad directly produces the modulation vector for the gradients, thus it does not require altering the architecture of the base network to achieve adaptation.

Remark 3. Meta-SGD benefits from the adaptive learning rate to accelerate its convergence and improve the performance of standard MAML. In Meta-SGD, the modulator is a global parameter for all tasks. Our method differs from Meta-SGD in that we use a task-dependent gradient modulation generated from neural networks.

Remark 4. In practice and to lower the computational complexity, one can make use of a set of distinct ModGrad cells $\boldsymbol{\Psi} = \{\boldsymbol{\psi}_1 \dots \boldsymbol{\psi}_m\}$, each acting on and

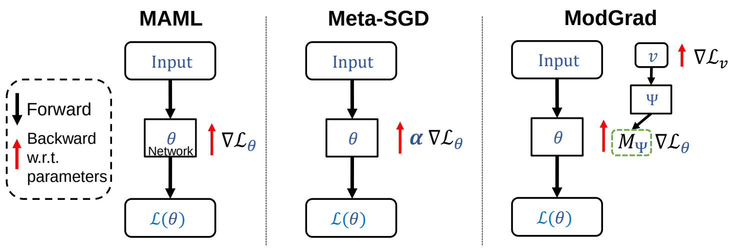

MAML
Meta-SGD
ModGrad
Forward
Backward w.r.t. parameters
Input
θ
Network
$\nabla\mathcal{L}_\theta$
$\alpha\,\nabla\mathcal{L}_\theta$
$\mathcal{L}(\theta)$
v
$\nabla\mathcal{L}_v$
Ψ
$M_\Psi\nabla\mathcal{L}_\theta$

Fig. 4: Comparison of our method to MAML [9] and Meta-SGD [22].

Method	Inner-Loop Update
MAML [9]	$\boldsymbol{\theta}^{(k+1)} \leftarrow \boldsymbol{\theta}^{(k)} - \alpha\nabla_{\boldsymbol{\theta}^{(k)}}\mathcal{L}(\boldsymbol{\theta}^{(k)})$
Meta-SGD [22]	$\boldsymbol{\theta}^{(k+1)} \leftarrow \boldsymbol{\theta}^{(k)} - \boldsymbol{\alpha}\nabla_{\boldsymbol{\theta}^{(k)}}\mathcal{L}(\boldsymbol{\theta}^{(k)})$
CAVIA [45]	$\boldsymbol{\nu}^{(k+1)} \leftarrow \boldsymbol{\nu}^{(k)} - \alpha\nabla_{\boldsymbol{\nu}^{(k)}}\mathcal{L}(\boldsymbol{\pi}, \boldsymbol{\nu}^{(k)}, \boldsymbol{\theta})$
LEO [33]	$\boldsymbol{z}^{(k+1)} \leftarrow \boldsymbol{z}^{(k)} - \alpha\nabla_{\boldsymbol{z}^{(k)}}\mathcal{L}(\boldsymbol{z}^{(k)}, \boldsymbol{\theta}, \boldsymbol{\phi}_e, \boldsymbol{\phi}_d, \boldsymbol{\phi}_r)$

Table 1: Comparison of inner-loop updates of various meta-learners. The red font shows additional parameters updated in the outer-loop.

optimizing a layer of the network (see Fig. 2 for a conceptual diagram). A ModGrad cell may be combined with any part of neural network including fully-connected and convolutional layers. This design requires no changes to the base network and brings flexibility, letting modulate the gradient for selected layers. Implementation details for fully-connected and convolutional layers are left in the supplementary material.

4 Related Work

Enjoying some theoretical guarantees, meta-learning by optimization (*e.g.*, [5, 14]) encompasses methods that learn the parameters of the model, at the meta-level, through gradient updates. The celebrated "model agnostic meta-learning" by Finn *et al.* [9] and "Learning to Learn" by Andrychowicz *et al.* [2] are prime examples of meta-learning by optimization. Note that, our meta-learning approach is an adaptive module acting on the model parameters which differs from relation and metric learning approaches *e.g.*, [17, 23, 35, 36, 42, 43].

Extensions of MAML include Reptile [26] that directly combines the updated parameters in the inner loop with that of the meta-learner, leading to a highly scalable meta-learning algorithm. MAML++ [3] uses a weighted loss in the inner-loop to boost the accuracy. We conclude this section by providing a brief overview on optimization-based meta-learning algorithms (see Table 1 for more details).

Preconditioned stochastic gradient descents (SGD). The general framework of preconditioning for fast adaptation of neural network is introduced in [11]. We discuss Meta-SGD and Meta-Curvature as the specific forms of preconditioned SGD. In Meta-SGD [22], the gradient update is equipped with a universal meta-parameter (α). In Meta-Curvature [27], the meta-parameter has a form of preconditioning matrices which transform the gradient in the inner-loop. During training, the meta-parameters of both methods [22, 27] change in each meta-update, however, they remain unchanged when the algorithm is applied to unseen tasks (see Fig. 4 for a conceptual diagram). Furthermore, additional meta-parameters are required if one requires to use more than one update in the inner-loop while ModGrad does not require extra parameters.

Network modulation. The modulation can be performed in the feature space [40, 45]. Borrowing the idea from FiLM [28], a function $f_{\boldsymbol{\pi}}(.)$ is designed to create the modulation for conditioning layers. In the inner-loop, CAVIA updates only the context vector ($\boldsymbol{\nu}$) which is the input of $f_{\boldsymbol{\pi}}$. The context vector is used as additional inputs to layers of the neural network to facilitate adaptation. In the same spirit, meta-transfer learning [37] learns the functions which scale and shift network parameters for each different task.

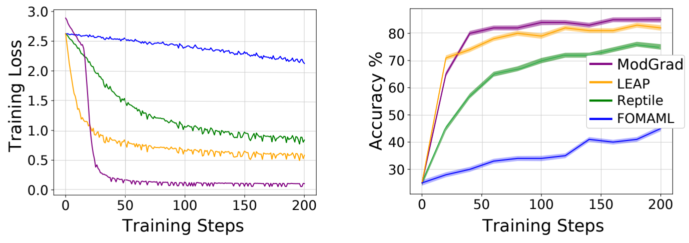
Network augmentation. Other approaches achieve meta-learning through augmenting the fast-adaptive network parameters in addition to the base networks. For instance, LEO [33], (M)T-Nets [21], and Meta-Networks [25] use some parts of the neural networks or generate them for fast-adaptation.

5 Experiments

In this section, we compare and contrast the proposed ModGrad method against baselines and state-of-the-art algorithms on various meta-learning tasks such as few-shot classification, image completion, and robot navigation. We conclude the section by an ablation study and analysis of the robustness of ModGrad in the presence of noisy gradients. Full details of all the experiments, including description of datasets, nuances of training, and hyperparameters can be found in the supplementary material of our paper. The source code of ModGrad is available at `https://github.com/chrysts/generative_preconditioner`.

5.1 Classification

Multi-shot classification. For multi-shot classification, we used the Omniglot dataset [19] containing 1623 characters from 50 different alphabets. The task is to classify images given a random number of sampled images (class-imbalance). Following the multi-shot setting in [10], we assess the rate of convergence and the performance of ModGrad in comparison to the LEAP [10], Reptile [26], and FOMAML [9] for a 20-way problem with 25 tasks in total (see Fig. 5 for details). We stress that in this experiment, the number of samples per class varies randomly between 1 and 25. Fig. 5 suggests that ModGrad performs adaptation much faster and to a much lower training loss while achieving the highest recognition accuracy of 85.1% compared to 83.3% of LEAP and 76.4% of Reptile. To the best of our knowledge, the performance of ModGrad for the multi-shot classification is the state of the art on this protocol and it tops previous studies by a notable margin.



(a) Training loss (b) Testing accuracy

Fig. 5: Training loss and testing accuracy on Omniglot in the multi-shot setting.

Few-shot classification. As our next experiment, we benchmark the ModGrad method for few-shot classification on the *mini*-ImageNet dataset [32]. In this comparison, we compare ModGrad to the related gradient-based meta-learning algorithms. The *mini*-ImageNet is a subset of the ImageNet [18] with 64, 16, and 20 classes for training, validation, and testing, respectively. We follow the widely-used protocol in the form of *episodes* for 5-way 5-shot and 1-shot with 600 tasks for testing. In all experiments, we performed episodic training using two network architectures, namely 4-convolutional blocks (Conv-4) without augmentation following [9, 36] and WideResNet 28-10 (WRN-28-10) [41] with augmentation following [29, 33].

As paper [30] suggests that the representations in last layers undergo significant changes during adaptation, we apply updates to the last two convolutional layers of the base network. The model parameters of the base networks and ModGrad are optimized with the Adam optimizer [16]. The learning rate is set to 10^{-3} and then cut by half for every 10K episodes. The size of $\boldsymbol{\nu}$ and value of α are set to 300 and 0.1 for all experiments on *mini*-ImageNet.

On this benchmark, ModGrad outperforms existing few-shot methods for various backbones *e.g.*, Conv-4 and WRN-28-10 as presented in Table 2. For a fair comparison, the result is compared to the meta-learning algorithms with the same backbones and experimental setup. Using Conv-4, ModGrad only needs 1-step and 64 filters per layer to outperform CAVIA, which employs 5-steps and 512 filters, by around 1.4% and 3.3% for 5-way 1-shot and 5-shot protocols. Furthermore, ModGrad with WRN-28-10 also performs better than the works by [20] and [33]. ModGrad needs only 1-step in the inner-loop compared to other meta-learning methods that need more than 1-step to achieve good results.

Number of steps. Below, we investigate meta-learning on deeper networks using ResNet [15] and a higher number of shots to capture the relationship between the number of step and these two factors. The *mini*-ImageNet dataset is used for experiments. To this end, we reimplement MAML and use the first-order method as the memory load for the second-order method is enormous given very deep networks. Note that the reported number of step is applied for both training and testing stages for the 5-way classification. To investigate the

Model	**Backbone**	**1-shot**	**5-shot**
ML LSTM [32]	Conv-4	43.44 ± 0.77	60.60 ± 0.71
MAML (64)$^{\#}$ [9]	Conv-4	47.89 ± 1.20	64.59 ± 0.88
Reptile [26]	Conv-4	49.97 ± 0.32	65.99 ± 0.58
Meta-SGD ([22]	Conv-4	50.50 ± 1.90	64.00 ± 0.90
R2-D2 [6]	Conv-4	48.70 ± 0.60	65.50 ± 0.60
(M)T-Net [21]	Conv-4	51.70 ± 1.84	–
CAVIA (512) [45]	Conv-4	51.82 ± 0.65	65.85 ± 0.55
ModGrad (1-step)	Conv-4	$\mathbf{53.20 \pm 0.86}$	$\mathbf{69.17 \pm 0.69}$
Qiao et al. [29]	WRN-28-10	59.60 ± 0.41	73.74 ± 0.19
MTL [37]	ResNet-12	61.20 ± 1.80	75.50 ± 0.80
LEO [33]	WRN-28-10	61.76 ± 0.08	77.59 ± 0.12
SCA + MAML++ [4]	DenseNet	62.86 ± 0.79	77.64 ± 0.40
wDAE-MLP [13]	WRN-28-10	62.67 ± 0.15	78.70 ± 0.10
wDAE-GNN [13]	WRN-28-10	62.96 ± 0.15	78.85 ± 0.10
Meta-Curvature [27]	WRN-28-10	64.40 ± 0.10	80.21 ± 0.10
ModGrad (1-step)	WRN-28-10	$\mathbf{65.72 \pm 0.21}$	$\mathbf{81.17 \pm 0.20}$

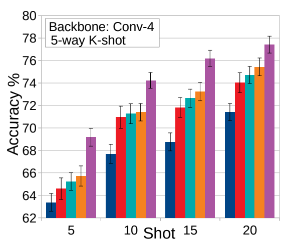
Table 2: The performance of existing gradient-based meta-learning methods for few-shot classification. Reported results are evaluated for 5-way 1- and 5-shot protocols on *mini*-ImageNet. MAML $^{\#}$ is our reimplementation.

relationship between the number of shot and the number of step, the number of shot is set to 5, 10, 15, and 20 samples using Conv-4 backbone. On ResNet-34, data augmentation and image size of 224×224 are used *without fine-tuning* the learning rate, following settings in [7]. Training using Conv-4 and ResNet-34 is performed over 50K and 100K episodes, respectively. Fig. 6 shows that MAML [9] needs more steps to achieve better results for higher shot number and deeper networks.

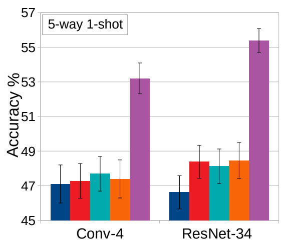
On the ResNet-34, we observe that the performance gap for 5-steps and 30-steps on Conv-4 (64 filters) is 2.5% but the performance gap reaches 4% on ResNet-34. Using ResNet-34, ModGrad outperforms MAML by ∼6.5% and ∼2.5% for 5-way 1-shot and 5-shot protocols. Using higher shot numbers, MAML with more additional steps also shows the improvement. The performance gap for 5-shot is about 2.5% between 5-steps and 30-steps but the performance gap increases up to 4% for 5-way 20-shot setting. Furthermore, in 1-step, ModGrad outperforms 30-steps by MAML by 2% in 20-shot classification. We conjecture that ModGrad achieves a good performance in 1-step for both cases because the modulation adaptively scales the gradients to reach a good minimum rapidly.

5.2 Regression

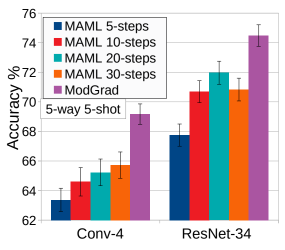
The task of image regression (completion) on the CelebA dataset [24] is adopted from [12]. The goal is to in-paint missing image pixels given only some pixels of images (random and ordered). For inputs pixel locations, models have to perform regression to approximate pixel intensities given 10 and 100 training pixels. The results in Table 3 show that ModGrad has the lowest Mean Square



(a)



(b)



(c)

Fig. 6: The performance of (a) ModGrad with various shot numbers, (b) deeper networks, and (c) MAML with 5, 10, 20, and 30 steps in the inner-loop. In 1-step, ModGrad achieves superior performance given various shots and backbones. MAML cannot perform well with a deeper network and larger shot numbers.

Error (MSE) on the image regression tasks compared to Conditional Neural Process (CNP) [12], CAVIA [45], and MAML [9]. We use the same setup as stated in [45] with five 128 hidden layers and a 128-dimensional input vector ($\boldsymbol{\nu}$). The learning rate is set 0.1. For this regression task, ModGrad is applied only to a single fully-connected layer (preceding the last layer) as we observed no tangible difference if applying ModGrad to several layers. Note that our results use only 1-step while CAVIA [45] and MAML [9] use 5 gradient steps to train from the training pixels. The qualitative results of image completion are shown in Fig. 7.

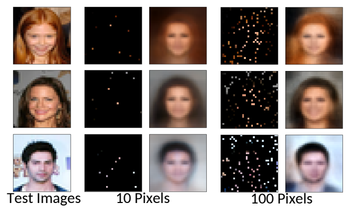


Fig. 7: Qualitative results of ModGrad on the CelebA dataset for image completion with 10 and 100 pixels provided randomly.

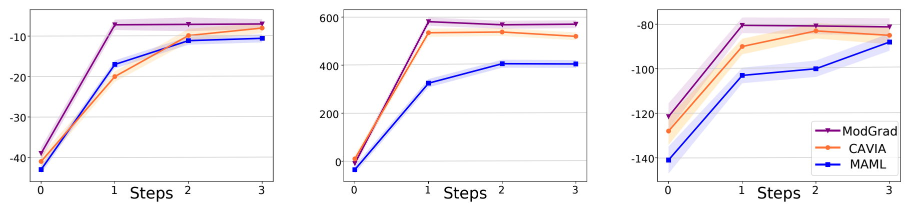
Model	Random Pixels		Ordered Pixels	
	10	**100**	**10**	**100**
CNP [12]	0.039	0.016	0.057	0.047
MAML [9]	0.040	0.017	0.055	0.047
CAVIA [45]	0.037	0.014	0.053	0.047
ModGrad	**0.034**	**0.012**	**0.048**	**0.043**

Table 3: The MSE for image completion tasks on the CelebA dataset for 10 and 100 pixels provided randomly and in the ordered fashion.

5.3 Reinforcement Learning

Our experiments on reinforcement learning are adopted from [9,45]. All network architectures, range of parameters, and protocols follow the same setup.

2D Navigation. In this experiment, we evaluate ModGrad on 2D-Navigation tasks from [9]. Every task contains a randomly chosen goal position where an agent has to move towards this position. The goal of this task is to adapt the policy of an agent quickly such that it can maximize the (negative) rewards.



(a) 2D Navigation (b) Half-Cheetah Dir (c) Half-Cheetah Vel

Fig. 8: Results for reinforcement learning on 2D navigation, half-cheetah direction, and velocity.

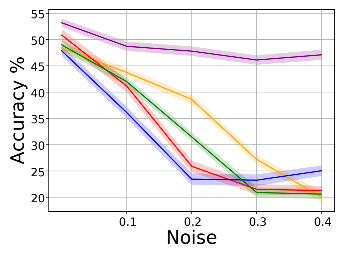
The goals of this navigation are within the range $[-0.5, 0.5]$ and the actions are clipped within $[-0.1, 0.1]$. In total, 20 trajectories are used for one gradient update. As in [45], we use the same network with two-layers, 100 hidden units, and a ReLU activation function. In 1-step, ModGrad achieves rewards around -8 while CAVIA and MAML are far below with -15.

Locomotion. We evaluate our method on the half-cheetah locomotion tasks from the MuJoCo simulator [38]. The tasks consist of predicting the direction and the velocity. The velocity ranges between 0.0 and 2.0. Each rollout length is 200, and 20 rollouts are used per gradient step during training. ModGrad reaches rewards around 590 with only 1-step but CAVIA and MAML obtain rewards below 550 only for half-cheetah direction tasks. Furthermore, ModGrad reaches around -80 for half-cheetah velocity tasks with 1-step but CAVIA and MAML reach only around -90 and -100, respectively.

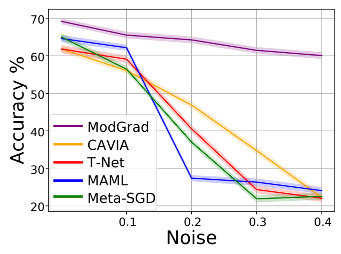
In all reinforcement learning tasks,] Fig. 8 shows that ModGrad requires fewer updates to achieve better rewards. This shows that our method is also beneficial for non-differentiable and dynamic problems.

5.4 How Robust is ModGrad to Noisy Gradients?

Methods such as CAVIA [45] and T-Net [21] modulate parameters or use an additional layer to transform gradients. These methods receive the gradients directly from the base network. Below, we empirically show that these approaches are fragile in the presence of corrupted gradients (which is a fundamental issue in the few-shot regime). To evaluate the robustness, we corrupted the gradients $\nabla\mathcal{L}(\boldsymbol{\theta})$ by adding noise following $\mathcal{N}(\mathbf{0}_n, \eta\mathbf{1}_n)$ to the gradients in the inner-loop, and we measured the accuracy of ModGrad, CAVIA, T-Net for various η (see Fig. 9 for results) using Conv-4 on *mini*-ImageNet. Compared to other methods, ModGrad degrades gracefully. For example, while our method only degrades about 10%, T-Net, CAVIA, Meta-SGD, and MAML plummet by 30% and 40% for 5-way 1-shot and 5-way 5-shot protocols, respectively.



(a) 5-way 1-shot



(b) 5-way 5-shot

Fig. 9: The performance comparison on *mini*-ImageNet with various noise level for 5-way 1-shot and 5-way 5-shot protocols.

5.5 Ablation Study

Below, we provide an ablation study on our proposed method using Conv-4 backbone. The experiments show how the number of steps and the column dimension of the tall matrix (u) used by the outer product affect the performance.

Impact of steps. We run 5-steps in inner-loop to check the performance of 5-way 5-shot and 1-shot protocols on *mini*-ImageNet suing Conv-4. Table 4 shows that ModGrad achieves a good performance in 1-step while running over more steps may vary the performance ±1% on 1-shot and 5-shot protocols. Thus, ModGrad is robust to a varying number of steps in few-shot learning.

Step	**1**	**2**	**3**	**4**	**5**
5-way 5-shot	69.17	68.80	68.42	68.34	68.24
5-way 1-shot	53.20	52.66	53.54	52.87	52.76

Table 4: ModGrad given various numbers of steps on *mini*-ImageNet.

The number of columns (u). Eq. 6 lets us choose the number of columns in the tall matrices. This experiment shows the impact of the number of columns (u) on results. Table 5 shows that the choice of u does not degrade the performance

Value of u	**1**	**5**	**10**
5-way 5-shot	68.43	69.17	67.62
5-way 1-shot	53.13	53.20	52.53

Table 5: The impact of column dimensions (u) on *mini*-ImageNet.

significantly ($\sim$0.5% variation on 5-way 1-shot protocol) but it may degrade results by $\sim$1.5% on the 5-way 5-shot protocol. In both cases, our approach outperforms the standard MAML (Conv-4) algorithm. We observe that the lower the value of u is the more the low-pass filtering nature of our gradient modulation is. With a full-rank matrix, the modulator looses its low-pass filtering nature. Our experiments on classification, regression and reinforcement learning use the value of $u = 5$.

5.6 Discussion

Time Complexity. ModGrad requires two forward and two backward passes per step. Thus, its complexity is 2× the computation of MAML but ModGrad converges significantly faster than MAML. For example, ModGrad with just one adaptation step comfortably outperforms MAML with five steps (best setting used by MAML). When comparing wall clocks, adaptation in MAML requires 0.05 second while ModGrad performs this step in 0.03 second per adaptation on the 5-way 5-shot protocol given Conv-4 backbone on mini-ImageNet.

Properties of ModGrad. We have observed that the low-rank modulating matrix paired with the Hadamard product acting on gradients have the property to perform adaptive low-pass filtering. This property depends on the context vector of the sister networks and the number of columns in the tall matrix controlled by the value of u. We believe this deems ModGrad a generative adaptive gradient filtering modulator which explains why ModGrad copes so well in the presence of gradients corrupted by the noise. Our supplementary material provides a more detailed theoretical analysis of the low-pass filtering properties of ModGrad. It also provides plots studying this property on both simulated and the real data.

6 Conclusions

This work presents a meta-learner by modulating the gradient via so-called ModGrad. Our approach shows a general ability to address a wide range of problems including few-shot classification, regression and reinforcement learning. Empirical results show that ModGrad is competitive compared with other existing gradient-based meta-learners. Furthermore, ModGrad is designed to be modular (applicable to every layer) in deep neural networks. Thus, it can be utilized for other interesting applications without any extra structural changes to the base network architecture. In practice, our approach copes with the practical optimization matters such as learning rates, gradient step and adaptation to noise. Our gradient-based meta-learning algorithm remains robust in the presence of corrupted gradients while other existing methods have a low tolerance. Another benefit of ModGrad is the accelerated learning of the base network. As a result, ModGrad works also well with deeper networks, higher number of shots and a lower number of steps compared to MAML.

References

1. Al-Shedivat, M., Bansal, T., Burda, Y., Sutskever, I., Mordatch, I., Abbeel, P.: Continuous adaptation via meta-learning in nonstationary and competitive environments. In: International Conference on Learning Representations (2018) 1
2. Andrychowicz, M., Denil, M., Gomez, S., Hoffman, M.W., Pfau, D., Schaul, T., Shillingford, B., De Freitas, N.: Learning to learn by gradient descent by gradient descent. In: Advances in neural information processing systems (2016) 7
3. Antoniou, A., Edwards, H., Storkey, A.: How to train your maml. In: International Conference on Learning Representations (2019) 1, 4, 7
4. Antoniou, A., Storkey, A.J.: Learning to learn by self-critique. In: Advances in Neural Information Processing Systems. pp. 9936–9946 (2019) 1, 10
5. Balcan, M.F., Khodak, M., Talwalkar, A.: Provable guarantees for gradient-based meta-learning. In: International Conference on Machine Learning. pp. 424–433 (2019) 7
6. Bertinetto, L., Henriques, J.F., Torr, P., Vedaldi, A.: Meta-learning with differentiable closed-form solvers. In: International Conference on Learning Representations (2019) 10
7. Chen, W.Y., Liu, Y.C., Kira, Z., Wang, Y.C.F., Huang, J.B.: A closer look at few-shot classification. In: International Conference on Learning Representations (2019) 10
8. Desjardins, G., Simonyan, K., Pascanu, R., et al.: Natural neural networks. In: Advances in Neural Information Processing Systems. pp. 2071–2079 (2015) 6
9. Finn, C., Abbeel, P., Levine, S.: Model-agnostic meta-learning for fast adaptation of deep networks. In: International Conference on Machine Learning (2017) 1, 3, 4, 7, 8, 9, 10, 11
10. Flennerhag, S., Moreno, P.G., Lawrence, N., Damianou, A.: Transferring knowledge across learning processes. In: International Conference on Learning Representations (2019) 8
11. Flennerhag, S., Rusu, A.A., Pascanu, R., Visin, F., Yin, H., Hadsell, R.: Meta-learning with warped gradient descent. In: International Conference on Learning Representations (2020) 7
12. Garnelo, M., Schwarz, J., Rosenbaum, D., Viola, F., Rezende, D.J., Eslami, S., Teh, Y.W.: Neural processes. arXiv preprint arXiv:1807.01622 (2018) 10, 11
13. Gidaris, S., Komodakis, N.: Generating classification weights with gnn denoising autoencoders for few-shot learning. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition. pp. 21–30 (2019) 10
14. Grant, E., Finn, C., Levine, S., Darrell, T., Griffiths, T.: Recasting gradient-based meta-learning as hierarchical bayes. In: International Conference on Learning Representations (2018) 7
15. He, K., Zhang, X., Ren, S., Sun, J.: Deep residual learning for image recognition. In: IEEE Conference on Computer Vision and Pattern Recognition (2016) 9
16. Kingma, D.P., Ba, J.L.: Adam: A method for stochastic optimization. In: International Conference on Learning Representations (2015) 9
17. Koniusz, P., Zhang, H.: Power normalizations in fine-grained image, few-shot image and graph classification. TPAMI (2020) 7
18. Krizhevsky, A., Sutskever, I., Hinton, G.E.: Imagenet classification with deep convolutional neural networks. In: Advances in neural information processing systems (2012) 9

19. Lake, B.M., Salakhutdinov, R., Tenenbaum, J.B.: Human-level concept learning through probabilistic program induction. Science **350**(6266), 1332–1338 (2015) 8
20. Lee, K., Maji, S., Ravichandran, A., Soatto, S.: Meta-learning with differentiable convex optimization. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition. pp. 10657–10665 (2019) 9
21. Lee, Y., Choi, S.: Gradient-based meta-learning with learned layerwise metric and subspace. In: International Conference on Machine Learning. pp. 2933–2942 (2018) 6, 8, 10, 12
22. Li, Z., Zhou, F., Chen, F., Li, H.: Meta-sgd: Learning to learn quickly for few shot learning. arXiv preprint arXiv:1707.09835 (2017) 2, 4, 7, 8, 10
23. Liu, Y., Lee, J., Park, M., Kim, S., Yang, E., Hwang, S., Yang, Y.: LEARNING TO PROPAGATE LABELS: TRANSDUCTIVE PROPAGATION NETWORK FOR FEW-SHOT LEARNING. In: International Conference on Learning Representations (2019) 7
24. Liu, Z., Luo, P., Wang, X., Tang, X.: Deep learning face attributes in the wild. In: Proceedings of International Conference on Computer Vision (December 2015) 10
25. Munkhdalai, T., Yu, H.: Meta networks. In: Proceedings of the 34th International Conference on Machine Learning-Volume 70. pp. 2554–2563. JMLR. org (2017) 8
26. Nichol, A., Achiam, J., Schulman, J.: On first-order meta-learning algorithms. arXiv preprint arXiv:1803.02999 (2018) 7, 8, 10
27. Park, E., Oliva, J.B.: Meta-curvature. In: Advances in Neural Information Processing Systems. pp. 3309–3319 (2019) 4, 8, 10
28. Perez, E., Strub, F., De Vries, H., Dumoulin, V., Courville, A.: Film: Visual reasoning with a general conditioning layer. In: Thirty-Second AAAI Conference on Artificial Intelligence (2018) 8
29. Qiao, S., Liu, C., Shen, W., Yuille, A.L.: Few-shot image recognition by predicting parameters from activations. In: IEEE Conference on Computer Vision and Pattern Recognition (2018) 9, 10
30. Raghu, A., Raghu, M., Bengio, S., Vinyals, O.: Rapid learning or feature reuse? towards understanding the effectiveness of maml. In: International Conference on Learning Representations (2020) 9
31. Rajeswaran, A., Finn, C., Kakade, S., Levine, S.: Meta-learning with implicit gradients. In: Advances in Neural Information Processing Systems (2019) 1
32. Ravi, S., Larochelle, H.: Optimization as a model for few-shot learning. In: International Conference on Learning Representations (2017) 1, 9, 10
33. Rusu, A.A., Rao, D., Sygnowski, J., Vinyals, O., Pascanu, R., Osindero, S., Hadsell, R.: Meta-learning with latent embedding optimization. In: International Conference on Learning Representations (2019) 1, 2, 4, 7, 8, 9, 10
34. Santoro, A., Bartunov, S., Botvinick, M., Wierstra, D., Lillicrap, T.: Meta-learning with memory-augmented neural networks. In: International conference on machine learning. pp. 1842–1850 (2016) 3
35. Simon, C., Koniusz, P., Nock, R., Harandi, M.: Adaptive subspaces for few-shot learning. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. pp. 4136–4145 (2020) 7
36. Snell, J., Swersky, K., Richard, Z.: Prototypical networks for few-shot learning. In: Advances in Neural Information Processing Systems (2017) 7, 9
37. Sun, Q., Liu, Y., Chua, T.S., Schiele, B.: Meta-transfer learning for few-shot learning. In: The IEEE Conference on Computer Vision and Pattern Recognition (2019) 1, 8, 10

38. Todorov, E., Erez, T., Tassa, Y.: Mujoco: A physics engine for model-based control. In: IEEE/RSJ International Conference on Intelligent Robots and Systems. pp. 5026–5033 (2012) 12
39. Vinyals, O., Blundell, C., Lillicrap, T., Kavukcuoglu, K., Wierstra, D.: Matching networks for one shot learning. In: Advances in Neural Information Processing Systems (2016) 3
40. Vuorio, R., Sun, S.H., Hu, H., Lim, J.J.: Multimodal model-agnostic meta-learning via task-aware modulation. In: Advances in Neural Information Processing Systems. pp. 1–12 (2019) 8
41. Zagoruyko, S., Komodakis, N.: Wide residual networks. In: Proceedings of the British Machine Vision Conference. pp. 87.1–87.12 (2016) 9
42. Zhang, H., Koniusz, P.: Power normalizing second-order similarity network for few-shot learning. In: Winter Conference on Applications of Computer Vision (2019) 7
43. Zhang, H., Zhang, L., Qui, X., Li, H., Torr, P.H.S., Koniusz, P.: Few-shot action recognition with permutation-invariant attention. ECCV (2020) 7
44. Zhang, Y., Qu, H., Chen, C., Metaxas, D.: Taming the noisy gradient: Train deep neural networks with small batch sizes. In: Proceedings of the Twenty-Eighth International Joint Conference on Artificial Intelligence, IJCAI-19. pp. 4348–4354 (2019) 1
45. Zintgraf, L., Shiarli, K., Kurin, V., Hofmann, K., Whiteson, S.: Fast context adaptation via meta-learning. In: International Conference on Machine Learning. pp. 7693–7702 (2019) 1, 4, 7, 8, 10, 11, 12