Memory Efficient Online Meta Learning

Durmus Alp Emre Acar [1] **Ruizhao Zhu** [1] **Venkatesh Saligrama** [1]

Abstract

We propose a novel algorithm for online meta learning where task instances are sequentially revealed with limited supervision and a learner is expected to meta learn them in each round, so as to allow the learner to customize a task-specific model rapidly with little task-level supervision. A fundamental concern arising in online meta-learning is the scalability of memory as more tasks are viewed over time. Heretofore, prior works have allowed for perfect recall leading to linear increase in memory with time. Different from prior works, in our method, prior task instances are allowed to be deleted. We propose to leverage prior task instances by means of a fixed-size state-vector, which is updated sequentially. Our theoretical analysis demonstrates that our proposed memory efficient online learning (MOML) method suffers sub-linear regret with convex loss functions and sub-linear local regret for nonconvex losses. On benchmark datasets we show that our method can outperform prior works even though they allow for perfect recall.

1. Introduction

Meta Learning (Vinyals et al., 2016; Santoro et al., 2016) is defined as a problem of learning to learn new tasks. This is typically accomplished by training a meta-model on a diverse set of tasks, such that the meta-model can in turn train and output a classifier on a new task using only a few training examples. There has been a burst of activity on meta-learning (Finn et al., 2017; Ravi & Larochelle, 2017; Zhou et al., 2019) recently with much of it devoted to the batch setting. Namely, during training, datasets composed of different tasks are provided, and the goal is to train a meta-learner, which at test-time can be rapidly re-purposed for a new task by training on relatively few annotated examples.

[1]Boston University, Boston, MA. Correspondence to: Durmus Alp Emre Acar <alpacar@bu.edu>, Ruizhao Zhu <rzhu@bu.edu>, Venkatesh Saligrama <srv@bu.edu>.

Proceedings of the 38^{th} International Conference on Machine Learning, PMLR 139, 2021.

While human learning is evidently the inspiration for traditional meta-learning approaches, there are significant differences. Humans learn continually by dynamically and continuously adapting their representations, beliefs and predictions as they encounter new tasks (Lake et al., 2017). Motivated by applications that involve such continual and lifelong learning, Finn et al. (2019) advocate an online meta-learning approach, in order to bridge the seeming gap between online learning and meta-learning. Their point is that online learning (Shalev-Shwartz & Singer, 2007) mirrors the way in which humans sequentially encounter new tasks one after another, but unlike humans, online learning does not allow for either task-specific adaptation or consider how solution to past tasks can help solve new ones. On the other hand, while conventional meta-learning does incorporate task-specific adaptation and incorporates knowledge transfer from past experiences, it ignores the episodic, sequential and non-stationary aspects of how tasks are encountered.

In this paper, we develop a novel method for online meta-learning overcoming drawbacks of Finn et al. (2019). Like Finn et al. (2019) we focus on training deep neural network models, and consider the setting where task instances are revealed to the learner episodically. We also attempt to train the model's initial parameters so that, on any new task, the model parameters can be rapidly adapted with a small amount of data by means of gradient descent. Subsequently, the learner then updates its underlying task-agnostic parameters based on solving the new task.

Linear Memory Scaling. Finn et al. (2019)'s follow-the-meta-learner (FTML) leverages the well-known follow the regularized leader (FTRL) method in online learning. While learning task-agnostic model parameters (meta-learning step), FTML recalls all task instances that have heretofore appeared. This is undesirable and impractical, and as such leads to linear increase in memory with the number of observed tasks. One option is to update the meta model only using the current task, but this leads to significant current task bias. Another possibility is to leverage an online gradient descent (OGD) algorithm, based on linearization of loss-functions for prior tasks, but the linearization would be around stale model parameters associated with the previous tasks. While practical, empirically these options do not lead to meaningful meta-learning performance.

Memory Efficient Online Meta-Learning (MOML). To overcome memory scaling, we introduce a fixed-size state-vector, which is dynamically updated after completion of an episodic task. The state-vector, which serves the purpose of encoding past task experiences, parameterizes the regularizer penalty for next episode. As a result, model parameters retain prior task experience, and utilize this experience to solve new tasks. Our MOML scheme is not only memory efficient, but is more effective than FTML. We compare MOML with FTML on current tasks as well as past tasks (to evaluate catastrophic forgetting), and show that MOML dominates FTML on several benchmark datasets. We also analyze MOML theoretically and show that for T tasks, we achieve sub-linear $O(\sqrt{T})$ regret on convex losses, and $O(\sqrt{T})$ local regret for non-convex loss functions.

Our Contributions.

- We present, MOML, a new family of online learning algorithms that do not explicitly store loss functions from previous rounds.
- We show that MOML has $O\left(\sqrt{T}\right)$ regret guarantees,
- We empirically show that MOML achieves significantly improved memory footprint with no perceptible degradation in performance over existing baselines.

1.1. Related Work

Meta learning is a popular and evolving field, and we only describe briefly different approaches (see Hospedales et al. (2020); Vanschoren (2018) for surveys). One approach is to learn a black box optimizer for the test task based on the state of the algorithm (Ravi & Larochelle, 2017; Mishra et al., 2017; Santoro et al., 2016). For instance, Ravi & Larochelle (2017) proposes an LSTM model in which the states of the LSTM determines the specific optimization procedure for the current task. More recently, Finn et al. (2017) introduced Model-Agnostic Meta Learning (MAML) approach. MAML proposes one step gradient descent on the task datasets and the goal is to learn a good task-agnostic initialization that works well for all tasks. Many works have introduced extensions to MAML including novel transformations (Li et al., 2017; Zhou et al., 2019) as well as improved training algorithms (Antoniou et al., 2018). There are works that propose non parametric meta adaptations (Snell et al., 2017; Vinyals et al., 2016). For instance, prototypical adaptation (Snell et al., 2017) is a metric based method where the task specific model is obtained using kNN with respect to the class prototypes. Additionally, Chen et al. (2020) show that pretraining of meta models before meta learning improves performance. Different from ours, meta learning in these works is viewed as an offline concept where training and test-time are separated.

Online learning. In vanilla online learning (Shalev-Shwartz & Singer, 2007), (possible adversarial) loss functions are sequentially revealed and the learner is trained as well as tested at each round. The agent aims to minimize cummulative regret that measures how well the algorithm performs compared to the best possible fixed model in hindsight. Online learning is a well established field and we refer to extensive studies for more information Hazan (2019); Shalev-Shwartz (2012). Online gradient descent (OGD) (Zinkevich, 2003) proposes to take a gradient descent step in each round using the current loss. Follow the Regularized Learner (FTRL) (Abernethy et al., 2008) minimizes a regularized version of all seen loss functions. Different from meta learning, the learner is expected to minimize the loss functions, and does not leverage insights from past experiences. In contrast, our focus is on online meta learning problem where the agent is expected to meta learn the revealed task instances.

Continual learning, (a.k.a lifelong learning) (Thrun & Pratt, 2012) is related to online learning, with particular focus on catastrophic forgetting. In this context, the agent is expected to do well on the seen tasks as well as efficiently learn new task instances (Chen & Liu, 2018). For instance, Learn-to-Grow framework (Li et al., 2019) avoids forgetting previous tasks by expanding the network architecture of the learner with upcoming tasks. Variational continual learning (Nguyen et al., 2018) is a method using variational inference on a set which has representative datapoints from the seen tasks. In contrast, our goal is to reduce the memory footprint, by allowing for the learner to delete *all* data instances for past tasks. Additionally, our goal is to derive regret guarantees as in online learning.

Online meta learning (Finn et al., 2019) proposes to fuse meta learning with online learning. In online meta learning, an agent is expected to meta learn tasks where the tasks are sequentially revealed. Prior works have proposed to leverage OGD and FTRL to the online meta learning problem (Zhuang et al., 2020; Finn et al., 2017). In the FTML (Finn et al., 2019) approach the goal is to learn initializations of model parameters (meta-model) so as to allow for quick adaptation to all of the previously viewed task instances based on taking a few gradient steps from the meta-model. For this reason, FTML must store data from all seen tasks to update its meta model. This means that the memory complexity of FTML linearly grows as new tasks arrive which is impractical. We propose a memory efficient approach, which does not require storing past past instances.

Subsequent works such as OSML (Yao et al., 2020) and FTML-VS (Yu et al., 2020) propose extensions to FTML. OSML is a pipeline that has multiple so called meta blocks to facilitate the learning of new tasks. FTML-VS aims to decrease the number of datapoints used during meta training as tasks arrive. Nevertheless, these methods still require storing datapoints for all seen tasks. Different from these works,

our goal is to bypass storing datapoints corresponding to seen tasks.

2. Method

The learner's goal is to train a meta model, chosen from a parameterized model with parameters $\boldsymbol{\theta} \in \mathbb{R}^d$ for the setting where new task instances are sequentially revealed at each round. Each task has a specific joint distribution denoted as $\mathcal{P}_t$ in which task features $\boldsymbol{x} \in \mathcal{X}$ and the corresponding labels $y \in \mathcal{Y}$ are drawn from it $(\boldsymbol{x}, y) \sim \mathcal{P}_t$. The agent has access to a limited supervised dataset $D_t = \{(\boldsymbol{x}_t^i, y_t^i)\}_{i=1}^{N_t}$ for each task in order to obtain a task specific model. We define task loss as the expected loss with respect to $\mathcal{P}_t$ as $f^t(\boldsymbol{\theta}) = E_{\{\boldsymbol{x},y\}\sim P_t} L\left((\boldsymbol{x}, y); \boldsymbol{\theta}\right)$ where L is the loss function that the model incurs on the data tuple $(\boldsymbol{x}, y)$. Our objective is to get a sublinear rate of the following regret statement,

$$R_T = \sum_{t=1}^{T} f^t \circ U^t(\boldsymbol{\theta}^t) - \min_{\boldsymbol{\theta}\in\mathbb{R}^d} \sum_{t=1}^{T} f^t \circ U^t(\boldsymbol{\theta}) \quad (1)$$

where U^t is the meta adaptation function that transforms the meta model into a task specific model using the limited supervised dataset D_t and the agent is compared against the best fixed meta learner that has access to all loss functions $\{f^t\}_{t=1}^T$ in hindsight.

Adaptation function. The regret statement depends on the transformation function U^t for task t. There are many transformations proposed in meta learning field to obtain a task specific model out of the meta model. In this work, we focus on MAML (Finn et al., 2017) adaptation. MAML transformation is proposed as,

$$U^t(\boldsymbol{\theta}) = \boldsymbol{\theta} - \eta \frac{1}{|D_t|} \sum_{(\boldsymbol{x},y)\in D_t} \nabla L\left((\boldsymbol{x}, y); \boldsymbol{\theta}\right)$$

and corresponds to updating the meta model with a step gradient descent using a meta learning rate η.

MOML Intuition. Before describing the algorithm, we build intuition for our solution by considering two conventional online learning algorithms: FTRL and OGD. Let us assume that we have $\{\ell^t\}_{t=1}^T$ losses in an online setting.

FTRL Algorithm. FTRL updates its model using all seen losses and a regularizer, namely,

$$\boldsymbol{\theta}^{t+1} = \arg\min_{\boldsymbol{\theta}} \ \frac{\mu}{2}\|\boldsymbol{\theta}\|^2 + \sum_{i=1}^{t} \ell^i(\boldsymbol{\theta}), \quad (2)$$

where μ is the coefficient on the quadratic regularizer. FTRL must store the history of all the past observed loss functions. Since the optimal competitor minimizes the sum of these losses (i.e. $\sum_{t=1}^T \ell^t(\boldsymbol{\theta})$), we can view FTRL, with proper regularization, converging to the optimal competitors risk at the expense of storing all seen task information.

OGD Algorithm. Different from FTRL, OGD does not store the seen losses. It applies one gradient descent step using the currently revealed loss as,

$$\boldsymbol{\theta}^{t+1} = \boldsymbol{\theta}^t - \beta \nabla \ell^t\left(\boldsymbol{\theta}^t\right) \quad (3)$$

where β is the learning rate. Even though it gets to desired regret guarantees, it is hard to compare OGD to the best competitor that minimizes the sum of losses (i.e. $\sum_{t=1}^T \ell^t(\boldsymbol{\theta})$).

The Bias Problem. OGD updates the model with gradients arising from the loss revealed at that time. Since, the best competitor seeks to optimize the sum of losses, one could learn the best competitor, by running SGD on the average of all losses as

$$\boldsymbol{\theta}^{k+1} = \boldsymbol{\theta}^k - \beta \frac{1}{T} \sum_{t=1}^{T} \nabla \ell^t\left(\boldsymbol{\theta}^k\right). \quad (4)$$

This iteration has the property that it converges to the optimal solution in hindsight for convex losses and a suitable choice of β.

The difference between OGD update in 3 and the competitor update in 4 is that the gradient directions are not aligned. More explicitly, the minima of the current loss is not the same as the competitor, $\min \ell^t(\boldsymbol{\theta}) \neq \min \sum_{s=1}^T \ell^s(\boldsymbol{\theta})$. As such the gradients have different directions. We propose a solution based on debiasing the gradient of the current loss, in the hope of taking a step towards the global gradient, while bypassing the need for recalling past seen instances.

A Toy Example. We illustrate the bias issue for an online learning setting with $T = 6$ losses where the parameter space is two dimensional. Figure 1 shows the contour plots (for quadratic loss functions) and the corresponding local optimal solutions for the 5 seen losses; the current revealed loss $\ell^6(\boldsymbol{\theta})$; and the sum of all losses $\sum_{t=1}^6 \ell^t(\boldsymbol{\theta})$. The learner's parameters at this time is depicted as $\times$.

Using the current loss, we can only go towards minima of the cuurent loss where the direction is shown with a red arrow, 'biased direction'. However, we would like to move towards the global minima denoted as the 'correct direction' since it points to the best competitor. The biased direction and the correct directions are not aligned which we refer as the bias problem.

The Debiasing Concept. Unfortunately, at a round $t < T$, all of the losses have not yet been revealed, and as such we can debias based only on the past losses. We propose to debias the current loss noted as 'debiasing' direction in Figure 1 by leveraging the past seen empirical loss functions. Then, we substitute a surrogate direction with the goal of correcting the biased direction. Our objective is to bypass

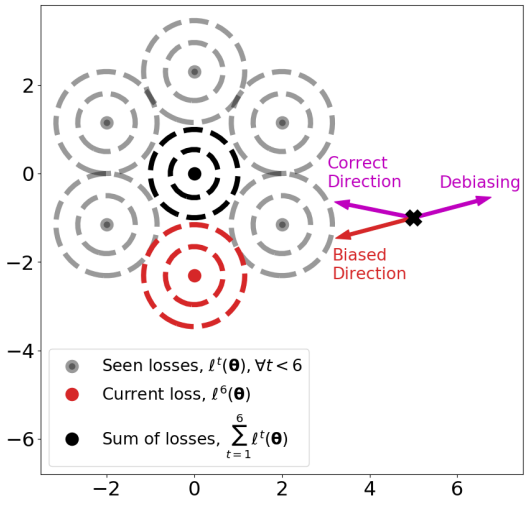

Figure 1. A two dimensional online learning setting with $T = 6$ losses. The current loss pulls the model towards its minima. However, the sum of losses have a different optimum point. Hence, the model is biased towards the current loss minima. We propose to debias the gradient so that the model is correctly updated.

the need to recall previous seen instances in computing this correction.

Let us denote the current model as $\boldsymbol{\theta}^t$. We start with the current model, $\boldsymbol{\theta}_1^{t+1} = \boldsymbol{\theta}^t$ and apply K corrected gradient descent steps as,

$$\boldsymbol{\theta}_{k+1}^{t+1} = \boldsymbol{\theta}_k^{t+1} - \beta \left(\nabla \ell^t \left(\boldsymbol{\theta}_k^{t+1} \right) - \boldsymbol{d}^t + \boldsymbol{c}^t \right) \tag{5}$$

where $k = 1, 2, \dots K$, $\boldsymbol{d}^t$ debiases the current loss and $\boldsymbol{c}^t$ encourages the correct direction. We note that our proposed solution does not require to store the losses as such $\boldsymbol{d}^t$ and $\boldsymbol{c}^t$ terms are updated using only the current loss.

MOML Algorithm. Our proposed method is presented in Algorithm 1. In each round, the current loss (f^t) along with adaptation function (U^t) is modified using a quadratic regularization and it is revealed to the algorithm. A task specific model is obtained using the transformation and the current meta model $\boldsymbol{\theta}^t$ as $\overline{\boldsymbol{\theta}}^t = U^t(\boldsymbol{\theta}^t)$. Then, the performance of the task specific model is recorded as $f^t(\overline{\boldsymbol{\theta}}^t) = f^t \circ U^t(\boldsymbol{\theta}^t)$.

We first update the meta model by optimizing using a quadratic penalty:

$$\mathcal{R}^t(\boldsymbol{\theta}) = -\langle \nabla f^{t-1} \circ U^{t-1}(\boldsymbol{\theta}^t), \boldsymbol{\theta} \rangle + \frac{\alpha}{2} \left\| \boldsymbol{\theta} - \boldsymbol{w}^t \right\|^2, \tag{6}$$

where $\nabla f^{t-1} \circ U^{t-1}(\boldsymbol{\theta}^t)$ and $\boldsymbol{w}^t$ are the states the model stores. The linear term debiases the current loss and the second term corrects the direction as in $\boldsymbol{d}^t$ and $\boldsymbol{c}^t$ terms defined in update Eq. 5.

Algorithm 1 Memory Efficient Online Meta Learning-MOML

Input: $T, \nabla f^0 \circ U^0(\boldsymbol{\theta}^1) = \boldsymbol{w}^1 = \boldsymbol{\theta}^1 = \mathbf{0}, \alpha, K, \beta$
for $t = 1, 2, \dots T$ **do**
  Output $\boldsymbol{\theta}^t$, reveal f^t and U^t, suffer $f^t \circ U^t(\boldsymbol{\theta}^t)$,
  $\mathcal{R}^t(\boldsymbol{\theta}) = -\langle \nabla f^{t-1} \circ U^{t-1}(\boldsymbol{\theta}^t), \boldsymbol{\theta} \rangle + \frac{\alpha}{2} \left\| \boldsymbol{\theta} - \boldsymbol{w}^t \right\|^2$,
  $\boldsymbol{\theta}_1^{t+1} = \boldsymbol{\theta}^t$,
  for $k = 1, 2, \dots K$ **do**
    $\boldsymbol{\theta}_{k+1}^{t+1} = \boldsymbol{\theta}_k^{t+1} - \beta \left(\nabla f^t \circ U^t(\boldsymbol{\theta}_k^{t+1}) + \nabla \mathcal{R}^t(\boldsymbol{\theta}_k^{t+1}) \right)$
  end for
  $\boldsymbol{\theta}^{t+1} = \boldsymbol{\theta}_{K+1}^{t+1}$,
  $\boldsymbol{w}^{t+1} = \frac{1}{2} \left(\boldsymbol{w}^t + \boldsymbol{\theta}^{t+1} - \frac{1}{\alpha} \nabla f^t \circ U^t(\boldsymbol{\theta}^{t+1}) \right)$,
end for

Subsequently, the algorithm iteratively optimizes the loss function with gradient corrections as,

$$\boldsymbol{\theta}_{k+1}^{t+1} = \boldsymbol{\theta}_k^{t+1} - \beta \left(\nabla f^t \circ U^t(\boldsymbol{\theta}_k^{t+1}) + \nabla \mathcal{R}^t(\boldsymbol{\theta}_k^{t+1}) \right). \tag{7}$$

After K gradient descent updates, the new meta model is obtained $\boldsymbol{\theta}^{t+1} = \boldsymbol{\theta}_{K+1}^{t+1}$.

MOML explicitly stores states $(\nabla f^{t-1} \circ U^{t-1}(\boldsymbol{\theta}^t), \boldsymbol{w})$ by way of summarization of previous task instances, and as such incorporates this information in the constructed regularizer.

$\boldsymbol{w}$ state is recursively updated as,

$$\boldsymbol{w}^{t+1} = \frac{1}{2} \left(\boldsymbol{w}^t + \boldsymbol{\theta}^{t+1} - \frac{1}{\alpha} \nabla f^t \circ U^t(\boldsymbol{\theta}^{t+1}) \right). \tag{8}$$

This completes one round of update mechanism for MOML.

Buffered-MOML (B-MOML). MOML can be extended to the situation where a fixed size buffer of previous task instances are also used. In this embodiment, we are allowed to store the latest B losses. For this scenario, the update rule for $\boldsymbol{\theta}$ is:

$$\boldsymbol{\theta}_{k+1}^{t+1} = \boldsymbol{\theta}_k^{t+1} - \beta \left(\nabla L_B^t(\boldsymbol{\theta}_k^{t+1}) + \nabla \mathcal{R}_B^t(\boldsymbol{\theta}_k^{t+1}) \right).$$

where $L_B^t(\boldsymbol{\theta}) = \frac{1}{B} \sum_{i=0}^{B-1} f^{t-i} \circ U^{t-i}(\boldsymbol{\theta})$ is the sum of last B losses and

$$\mathcal{R}_B^t(\boldsymbol{\theta}) = -\left\langle \frac{1}{B} \sum_{i=0}^{B-1} \nabla f^{t-i-1} \circ U^{t-i-1}(\boldsymbol{\theta}^{t-i}), \boldsymbol{\theta} \right\rangle + \frac{\alpha}{2} \left\| \boldsymbol{\theta} - \boldsymbol{w}^t \right\|^2$$

is the adapted regularizer. We utilize B-MOML in deriving regret bounds for nonconvex (adversarial) setting.

Random Task Buffer. In experiments we also consider storing random tasks instances in our buffer. To do so, we consider a buffer as first-in-first-out (FIFO) queue. For each

new task, we sample a biased coin with parameter p. We accept the new task in the buffer and put it at end of our queue, and then delete the first task.

Memory footprint of MOML does not grow over time. Our proposed method does not require storing all seen task instances. Instead, it accumulates previous task information in the auxiliary model ($\boldsymbol{w}$). Different from MOML, FTML needs to increase its memory usage in each round since it explicitly stores previous task information. We note that this leads to significant savings in terms of memory requirement.

MOML can leverage any meta adaptation function (U^t). We can use any adaptation function such as MAML, or other objectives such as prototypical adaptation, etc. MOML leads to a new family algorithms that can be applied to any online learning setup.

2.1. Analysis of MOML

MOML minimizes the following risk in K gradient steps,

$$\min_{\boldsymbol{\theta}\in\mathbb{R}^d} f^t \circ U^t(\boldsymbol{\theta}) + \mathcal{R}^t(\boldsymbol{\theta}) \tag{9}$$

To simplify the convergence analysis, we assume that MOML reaches a stationary point of Eq. 9, namely,

$$\nabla f^t \circ U^t(\boldsymbol{\theta}^{t+1}) - \nabla f^{t-1} \circ U^{t-1}(\boldsymbol{\theta}^t) + \alpha\left(\boldsymbol{\theta}^{t+1} - \boldsymbol{w}^t\right) = \mathbf{0}. \tag{10}$$

This assumption is not unrealistic. Indeed, due to our quadratic penalty, in experiments, we have found that MOML is essentially reaches a stationary point (i.e., Eq. 10) within a small number of gradient steps. In particular, we find that the residual noise is significantly smaller than the model parameters, and can be ignored for the purpose of analysis. Nevertheless, we point out that it is possible to extend our results to include this additional residual noise.

MOML proposes to reach a similar solution without explicitly storing losses from previous rounds. We present an intuitive justification for our viewpoint based on the assumption that MOML is a stable[1] algorithm and the sequential updates, $\boldsymbol{\theta}^t$, converge, namely, $\lim_{t\to\infty} \boldsymbol{\theta}^t = \boldsymbol{\theta}'$. Nevertheless, with this assumption in place, it follows that MOML, if it converges, converges to a stationary point of the desired loss function. We state this as a proposition.

Proposition 1 *Suppose $f^t \circ U^t(\cdot)$ are a sequence of smooth functions with uniformly bounded Lipshitz constant. Furthermore, suppose $\boldsymbol{\theta}^t$ is a bounded convergent sequence approaching $\boldsymbol{\theta}'$. Then, $\boldsymbol{\theta}'$ is also a stationary point of the competitor defined as in Eq. 1.*

[1]Validating this assumption requires additional proof, such as showing the map $\boldsymbol{\theta}^t \to \boldsymbol{\theta}^{t+1}$ is contractive. In online meta learning, $\boldsymbol{\theta}^t$ convergence is not required since the losses are non-stochastic. The convergence happens in stochastic loss settings.

We sketch the proof below. Using Cesaro[2] mean argument, if models converge, the mean model does as well: $\frac{1}{t}\sum_{s\in[t]} \boldsymbol{\theta}^s \underset{t\to\infty}{\longrightarrow} \boldsymbol{\theta}'$. If we average Eq. 10 over time, we observe that $\nabla f^t \circ U^t(\boldsymbol{\theta}^{t+1})$ terms telescope and we get $\frac{1}{t}\sum_{s\in[t]} \boldsymbol{\theta}^{s+1} - \frac{1}{t}\sum_{s\in[t]} \boldsymbol{w}^s = -\frac{1}{\alpha t}\nabla f^t \circ U^t(\boldsymbol{\theta}^{t+1})$. If the gradients are bounded we can assume $-\frac{1}{\alpha t}\nabla f^t \circ U^t(\boldsymbol{\theta}^{t+1}) \underset{t\to\infty}{\longrightarrow} \mathbf{0}$. Consequently we see that mean model and mean $\boldsymbol{w}$ state converge to the same model as $\frac{1}{t}\sum_{s\in[t]} \boldsymbol{w}^s \underset{t\to\infty}{\longrightarrow} \boldsymbol{\theta}'$. If we average update rule of $\boldsymbol{w}$ in Eq. 8 over time and plug these relations we get $\frac{1}{t}\sum_{s\in[t]} \nabla f^{s-1} \circ U^{s-1}(\boldsymbol{\theta}^s) \underset{t\to\infty}{\longrightarrow} \mathbf{0}$. Since we assume $\boldsymbol{\theta}^t \underset{t\to\infty}{\longrightarrow} \boldsymbol{\theta}'$, for sufficiently large t, $\frac{1}{t}\sum_{s\in[t]} \nabla f^{s-1} \circ U^{s-1}(\boldsymbol{\theta}') \underset{t\to\infty}{\longrightarrow} \mathbf{0}$. This is the stationary point relation of the accumulated losses. Therefore, MOML can be close to the optimal competitor without actually storing the losses from previous rounds.

As in standard online learning, we assume the losses to have a bounded gradient $\|\nabla f^t \circ U^t(\boldsymbol{\theta})\| \le G$. We first give a regret statement for convex losses.

Theorem 1 *For convex possibly adversarial $\{f^t \circ U^t\}_{t=1}^T$ functions and $\alpha = O\left(\sqrt{T}\right)$, Algorithm 1 satisfies,*

$$\sum_{t=1}^T f^t \circ U^t(\boldsymbol{\theta}^t) - \sum_{t=1}^T f^t \circ U^t(\boldsymbol{\theta}^*) = O\left(\sqrt{T}\left(\|\boldsymbol{\theta}^*\|^2 + G^2\right)\right),$$

where $\boldsymbol{\theta}^* = \arg\min_{\boldsymbol{\theta}\in\mathbb{R}^d} \sum_{t=1}^T f^t \circ U^t(\boldsymbol{\theta})$.

Theorem 1 gives sub-linear, $O\left(\sqrt{T}\right)$, regret rate for convex functions with bounded gradients. The dependency on T is optimal since there exists a setting where any algorithm suffers $\Omega\left(\sqrt{T}\right)$ regret for convex adversarial losses (Hazan et al., 2007).

Nonconvex Adversarial. It is well-known (see Hazan et al. (2017)) that for adversarial nonconvex losses, sub-linear regret for the formulation in Eq. 1 is generally difficult to achieve. One option is to instead evaluate $\sum_{t=1}^T \left\|\nabla \ell^t(\boldsymbol{\theta}^t)\right\|^2$, but this is not meaningful, since we want to reach a stationary point for the sum of the losses, and not each individual loss. As a tractable regret notion, Hazan et al. (2017) propose to use

$$\sum_{t=1}^T \left\| \frac{1}{B} \sum_{i=0}^{B-1} \nabla \ell^{t-i}(\boldsymbol{\theta}^t) \right\|^2, \tag{11}$$

where we consider a B sized window of the losses. This introduced time window smoothens the gradient of loss

[2]If a sequence $\{a_i\}_{i=1}$ converges $a_i \underset{i\to\infty}{\longrightarrow} a'$, mean also converges $\frac{1}{i}\sum_{s=1}^i a_s \underset{i\to\infty}{\longrightarrow} a'$.

suffered and leads to sub-linear regret. Similar to Hazan et al. (2017) algorithm, we use B-MOML to tackle this type of regret notion. B-MOML gets to $O\left(\frac{T}{B^2}G^2\right)$ regret. Hazan et al. (2017) show a lower bound where a set of losses of losses is constsructed in a way that any algorithm suffers $\Omega\left(\frac{T}{B^2}\right)$. Based on this result, our regret rate is optimal in terms of T and B dependencies. We give formal statement and the proof in Appendix A.4.

Stochastic Setting with Nonconvex Losses. As another extension, we propose a different regret notion for nonconvex losses where we do not need to consider a time window either in the regret statement or in the algorithm. In this notion, at each time, we assume the losses are chosen uniformly at random without replacement from a predetermined set of losses, $i_t \sim [K]$, $\{\ell^j\}_{j=1}^K$ and we aim to minimize

$$\sum_{t=1}^{T} E\left\|\frac{1}{K}\sum_{k=1}^{K}\nabla\ell^k(\boldsymbol{\theta}^t)\right\|^2,$$

where the expectation is with respect to the random index i_t of the losses.

Theorem 2 *Suppose $\mathcal{T}$ is a collection of tasks, and for each $\tau \in \mathcal{T}$, $f^\tau \circ U^\tau$ is nonconvex L smooth. We choose tasks i_t from some task distribution, $P_{\mathcal{T}}$ in an IID fashion. Then, for $\alpha = O\left(\sqrt{T}\right)$, Algorithm 1 satisfies,*

$$\sum_{t=1}^{T} E\left\|E_\tau[\nabla f^\tau \circ U^\tau(\boldsymbol{\theta}^t)]\right\|^2 = O\left(\sqrt{T}\left(\Delta + G^2 L\right)\right)$$

where $\Delta = E_\tau[f^\tau \circ U^\tau(\boldsymbol{\theta}^1)] - \min_{\boldsymbol{\theta}} E_\tau[f^\tau \circ U^\tau(\boldsymbol{\theta})]$.

Theorem 2 gives sub-linear regret rate for a nonconvex losses. For the stochastic setting, our regret rate is $O\left(\sqrt{T}\right)$.

Stochastic Setting with Nonconvex Polyak- Lojasiewicz (PL) Losses. The regret statement in Theorem 2 is motivated from the first order condition of stationary points. A better statement with respect to the optimum competitor can be obtained for PL nonconvex losses in the stochastic setting.

Corollary 1 *If each loss in Theorem 2 satisfies the (PL) condition, with parameter μ, it follows that,*

$$\sum_{t=1}^{T} E\left[E_\tau[f^\tau \circ U^\tau(\boldsymbol{\theta}^t)] - \min_{\boldsymbol{\theta}} E_\tau[f^\tau \circ U^\tau(\boldsymbol{\theta})]\right] = O\left(\sqrt{T}\frac{1}{\mu}\left(\Delta + G^2 L\right)\right).$$

Corollary 1 gives sub-linear rate with respect to the best competitor. We note that PL condition allows us to get a similar regret statement as in the convex setting even for nonconvex losses.

3. Experiments

This section displays our empirical comparison of MOML against competing baselines on standard datasets. We highlight main advantages of our method under various settings. We use PyTorch framework (Paszke et al., 2019) to train and evaluate our models. MAML meta training is implemented with Higher (Grefenstette et al., 2019) library. Hyperparameter tuning in an online setting poses challenges, since unlike the batch setting, we typically do not have a validation set. To overcome this issue we leverage the Hedge algorithm (Freund & Schapire, 1997) for hyperparameter tuning. We start by explaining the datasets, models and the baselines used for evaluations. We refer to Appendix A.1 for details of our setup.

Datasets. We evaluate the performance of our approach on three benchmark datasets: *MNIST* (LeCun et al., 1998), *CIFAR-100* (Krizhevsky et al., 2009) and *miniImageNet* (Vinyals et al., 2016). We state the task generation process for each of the datasets below.

Sequential MNIST (S-MNIST): Similar to Rainbow MNIST (Finn et al., 2019), we construct S-MNIST dataset consisting of diverse MNIST classification tasks. We generate a large-scale dataset consisting of 1000 tasks where train/test set of each task include transformations such as rotations, axes flip, cropping and scaling. These transformations are applied to class instances of each task. We note that unlike Rainbow MNIST (Finn et al., 2019) where a specific transformation is used in each task, we allow a more diverse tasks by including different transformations among classes within each task.

5 way-CIFAR-100: Similar to Finn et al. (2019), we construct a sequence of 5-way classification tasks within CIFAR-100 dataset. There are overall 200 tasks and train/test set of each task contains a 5-class combination from the 100 classes. Since only 5 classes are chosen out of 100 classes for each task, this generation ensures that the tasks are diverse, and particularly challenging for the online meta learning setting.

5 way-miniImagenet: miniImageNet dataset is collected as a subset ILSVRC-2012 (Deng et al., 2009) where there are a total of 100 classes with 600 images in each class. miniImageNet has realistic RGB images and it is harder compared to CIFAR100 dataset. Similar to 5 way-CIFAR-100, in 5 way-miniImageNet, we generate 100 tasks where each task has 5 classes from miniImageNet dataset.

Realistic Test-Bed. We note that S-MNIST and 5-way CIFAR-100 tasks are harder than the corresponding ones in Finn et al. (2019). Firstly, our setting is large scale where there are 1000 and 200 tasks for S-MNIST and 5 way-CIFAR-100 respectively compared to 60 and 50 tasks in prior work. Secondly, we have fewer number of training data per task in our setting, and in particular, 60 and 250

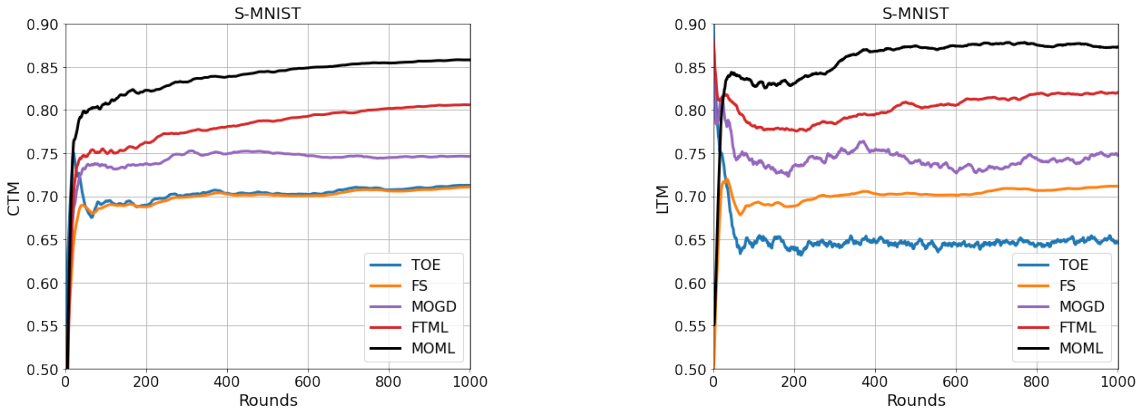

Figure 2. Experiment results on S-MNIST. **Left**: CTM versus rounds plot. MOML adapts well to the unseen tasks compared to the baselines. **Right**: Smoothed LTM versus rounds. MOML is robust to catastrophic forgetting.

training datapoints for S-MNIST and 5 way-CIFAR-100 datasets respectively. In contrast, Finn et al. (2019) has 900 and 2000 datapoints. We note that in online meta learning, we must leverage common knowledge across different tasks, since we do not have sufficient data for any one task. Our task generation resembles a more realistic and large-scale experimental test-bed. As a result, FTML performance is significantly lower than that reported in prior work.

Models. We use fully connected network architecture for S-MNIST experiments. The model takes flattened version of the image and passes through one hidden layer of size 200 neurons with ReLU non linearity followed by the softmax layer. For 5 way-CIFAR-100, we use a CNN architecture consisting of two convolutional layers with 64 5×5 filters, two max pooling layers, two fully connected layers with hidden sizes as 384 and 192 and a final output layer. In 5 way-miniImageNet, we use a similar CNN architecture where we have three convolutional blocks and max pooling layers followed by two fully connected layers of size 400 and 100 and a final softmax layer.

Methods. We compare MOML algorithm against two types of baselines. First set of baselines meta learns tasks such as FTML (Finn et al., 2019) and Meta OGD (MOGD). As described, FTML is an extension of FTRL in online meta learning setting where the algorithm stores all seen task instances. Similarly, we define Meta OGD (MOGD) where the algorithm extends OGD (Zinkevich, 2003) using the meta losses at each iteration. Different from FTRL, MOGD does not store losses. Our second set of baselines follows Finn et al. (2019) and includes train on everything (TOE) and train from scratch (FS). In TOE baseline, we do not meta learn a model, instead we store all tasks and learn one predictive model. During inference first fine tune TOE baseline using MAML and then record its performance. In FS baseline, we adapt a random model to each task using the limited data.

Performance Metrics. We report performance on three different metrics. These are Current Task Metric, Long-Term Task Metric and Task Learning Efficiency Metric.

Current Task Metric (CTM). We evaluate our models with respect to the current revealed task instance. We first allow the meta model to adapt to the current task using MAML adaptation and record its performance on task test data. We note that meta model is adapted directly before being trained on the task similar to the regret statement (Eq. 1).

Long-Term Task Metric (LTM). Different from CTM, we also look at the performance with respect to the previous tasks. At each round, we adapt the current meta model to each of the previous tasks using the limited data and record the performances with task test data. Then, we consider the average performance among the seen tasks. This metric measures *catastrophic forgetting*, and our goal is to ensure that past experiences are not forgotten. We note that the meta model is first adapted to the old tasks using associated training data in evaluating catastrophic forgetting and LTM.

Task Learning Efficiency Metric. Similar to Finn et al. (2019), we record the number of datapoints required to achieve a sufficient performance on the current task instance.

Analysis and Discussions. We report average CTM as well as LTM accuracy for all methods in Table 1. We present CTM and LTM versus rounds plots in Figure 2, 3 and 4 for S-MNIST, 5 way-CIFAR-100 and 5 way-miniImageNet settings respectively. We further test the effect of number of classes in each task on CIFAR-100 dataset and report the performances of MOML and FTML for the setting where each task has 3, 4 or 5 classes in Table 2.

MOML adapts well to unseen tasks, improving upon competing methods. MOML gets to similar or higher accuracy in CTM compared to the baselines in Table 1 and 2. For instance, in 5 way-CIFAR-100 setting, MOML reaches 5%

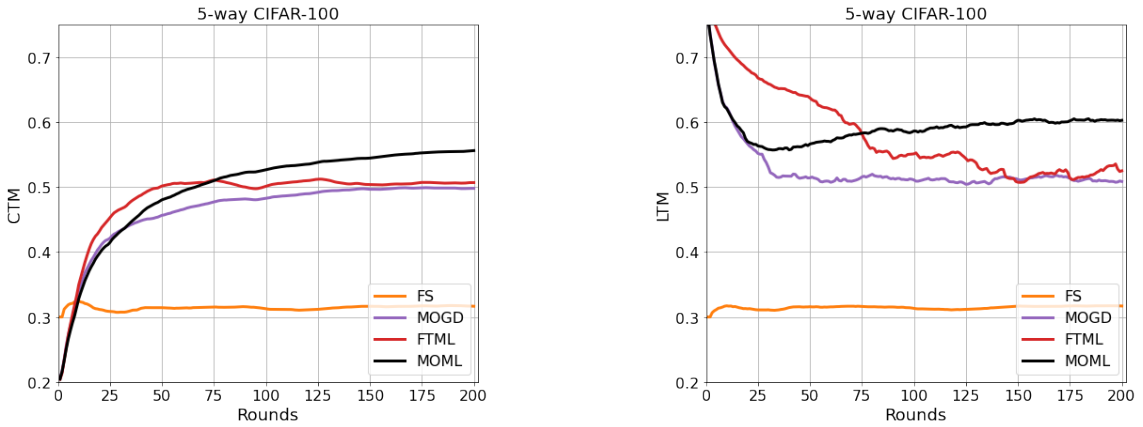

Figure 3. Experiment results on 5 way-CIFAR-100. **Left**: CTM versus rounds plot. MOML adapts well to the unseen tasks compared to the baselines. **Right**: Smoothed LTM versus rounds plot. MOML is robust to catastrophic forgetting.

Table 1. Performances for S-MNIST, 5way-CIFAR-100 and 5way-miniImageNet. TOE is not tabulated for 5way-CIFAR-100 and 5way-miniImageNet due to its poor performance.

	Test Accuracy	
S-MNIST	**CTM**	**LTM**
TOE	71.22	63.73
FS	71.15	71.14
MOGD	74.63	74.07
FTML	80.62	82.49
MOML	85.82	87.49
5 way-CIFAR-100		
FS	31.58	31.60
MOGD	49.92	50.21
FTML	50.68	54.50
MOML	55.83	60.78
5 way-miniImageNet		
FS	20.90	20.81
MOGD	49.08	56.86
FTML	56.77	63.12
MOML[3]	56.23	64.27

higher accuracy than FTML. As another example, Figure 2 shows that MOML strictly dominates all baselines after 100 rounds. These findings show that MOML can easily adapt new task instances.

MOML is task efficient. MOML achieves 80% test accuracy on new task instances by using less amount of limited data compared to FTML in S-MNIST as shown in Appendix A.1. The findings show that MOML is more task efficient and it can easily adapt to new task instances.

MOML is robust to catastrophic forgetting. MOML implicitly encodes past experience without explicitly storing it.

[3]For miniImageNet, MOML performances reported in Table 1 is B-MOML with buffer of 10 tasks.

Table 2. MOML and FTML performances for $3, 4$ and 5 way-CIFAR-100 settings.

Test Accuracy	**K-way**		
CTM	**3**	**4**	**5**
FTML	67.07	58.19	50.68
MOML	69.10	60.85	55.83
LTM			
FTML	61.84	54.70	54.50
MOML	72.15	62.79	60.78

This is evident in Table 1 and 2. For instance, in S-MNIST setting, MOML improves 5% LTM over FTML baseline. This shows that unlike competitors, MOML has superior performance on previously observed tasks.

MOML decreases memory complexity. Among the meta learning methods, MOML and MOGD do not store seen task instances. Different from these methods, FTML remembers all seen task information and minimizes the accumulated sum of losses. Not storing task instances for MOGD leads to performance degradation in comparison to FTML (Table 1). On the other hand, MOML outperforms both methods without storing previous data.

Meta learning is necessary for good generalization. TOE baseline learns one predictive model for all tasks. Since tasks are diverse, its performance is strictly lower than the methods that meta learns tasks even though we allow fine tuning during inference time. Similarly, FS does not use meta learning and directly adapts the model for each task. it performs poorly on both CTM and LTM (see Table 1).

Ablation study on B-MOML. B-MOML, a variant of MOML algorithm, described in Section 2, allows for storing B tasks in a buffer. We test B-MOML to see the effect of buffer size. Table 3 displays B-MOML performance with buffer sizes as $0, 5, 10$. We note that $B = 0$ corresponds to

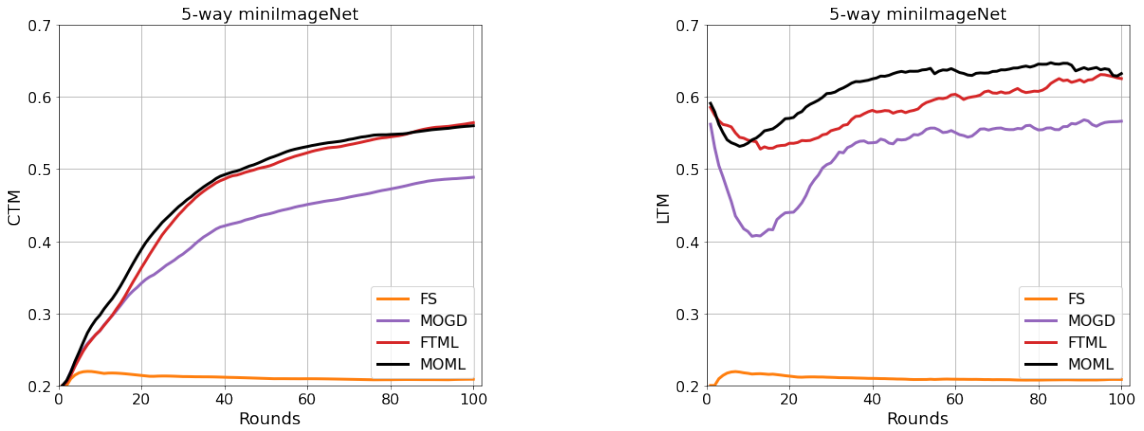

Figure 4. Experiment results on 5 way-miniImageNet. **Left**: CTM versus rounds plot. MOML adapts well to the unseen tasks compared to the baselines. **Right**: Smoothed LTM versus rounds plot. MOML is robust to catastrophic forgetting.

Table 3. Ablative study of B-MOML with $0, 5$ and 10 size buffers where $B = 0$ is original MOML.

Test Accuracy	**Buffer Size,** B		
S-MNIST	**0**	**5**	**10**
CTM	85.82	84.21	84.33
LTM	87.49	87.54	87.77
5 way-CIFAR-100			
CTM	55.83	56.37	56.70
LTM	60.78	63.76	65.24

original MOML algorithm where we do not store previous tasks.

MOML and B-MOML performances are comparable. We observe that storing seen task instances improves performance on new task and reduces the catastrophic forgetting in Table 3. In particular, LTM performances are improved with B-MOML. However, the change in CTM performance is marginal. For instance, increasing buffer from 0 to 10 increases CTM by only 1% in CIFAR-100 . We can infer that MOML effectively summarizes past task information.

Latest B-buffer is more effective than random B-buffer. As described in Section 2, we consider B-MOML variants where we allow buffers with the latest-B and a random variant of a fixed size. Table 4 shows that latest-B Buffer is somewhat more effective than the random case for MOML.

4. Conclusion

We propose a novel method MOML for online meta learning problem where an artificial agent is expected to meta learn sequentially coming tasks. We point out a practical concern that the agent can not store data from previously seen tasks as its memory footprint linearly grows with time. Different from the prior work, MOML stores local states

Table 4. Ablative study of buffer storing schemes of B-MOML.

Test Accuracy	**Buffer Size,** B	
S-MNIST	**5**	**10**
CTM		
Random	83.30	84.07
Last	84.21	84.33
LTM		
Random	86.09	86.97
Last	87.54	87.77
5 way-CIFAR-100		
CTM		
Random	56.40	55.07
Last	56.37	56.70
LTM		
Random	64.37	64.37
Last	63.76	65.24

that extract the experience information from the seen task. These local states significantly improve the memory complexity of MOML as we only store the states rather than all accumulated task data. We theoretically prove sub-linear regret rates and empirically show that MOML leads to huge memory savings by retaining the same or better performance compared to the competitors.

Acknowledgements

This research was supported by a gift from the ARM corporation, and by Army Research Office Grant W911NF2110246, the National Science Foundation grants CCF-2007350 (VS), CCF-2022446(VS), CCF-1955981 (VS), the Office of Naval Research Grant N0014-18-1-2257.

References

Abernethy, J., Hazan, E., and Rakhlin, A. Competing in the dark: An efficient algorithm for bandit linear optimization. In *21st Annual Conference on Learning Theory, COLT 2008*, pp. 263–274, 01 2008.

Antoniou, A., Edwards, H., and Storkey, A. How to train your maml. *arXiv preprint arXiv:1810.09502*, 2018.

Chen, Y., Wang, X., Liu, Z., Xu, H., and Darrell, T. A new meta-baseline for few-shot learning. *arXiv preprint arXiv:2003.04390*, 2020.

Chen, Z. and Liu, B. Lifelong machine learning. *Synthesis Lectures on Artificial Intelligence and Machine Learning*, 12(3):1–207, 2018.

Deng, J., Dong, W., Socher, R., Li, L.-J., Li, K., and Fei-Fei, L. Imagenet: A large-scale hierarchical image database. In *2009 IEEE conference on computer vision and pattern recognition*, pp. 248–255. Ieee, 2009.

Finn, C., Abbeel, P., and Levine, S. Model-agnostic meta-learning for fast adaptation of deep networks. In Precup, D. and Teh, Y. W. (eds.), *Proceedings of the 34th International Conference on Machine Learning*, volume 70 of *Proceedings of Machine Learning Research*, pp. 1126–1135, International Convention Centre, Sydney, Australia, 06–11 Aug 2017. PMLR. URL http://proceedings.mlr.press/v70/finn17a.html.

Finn, C., Rajeswaran, A., Kakade, S., and Levine, S. Online meta-learning. In Chaudhuri, K. and Salakhutdinov, R. (eds.), *Proceedings of the 36th International Conference on Machine Learning*, volume 97 of *Proceedings of Machine Learning Research*, pp. 1920–1930, Long Beach, California, USA, 09–15 Jun 2019. PMLR. URL http://proceedings.mlr.press/v97/finn19a.html.

Freund, Y. and Schapire, R. E. A decision-theoretic generalization of on-line learning and an application to boosting. *Journal of Computer and System Sciences*, 55(1):119–139, 1997. ISSN 0022-0000. doi: https://doi.org/10.1006/jcss.1997.1504. URL https://www.sciencedirect.com/science/article/pii/S002200009791504X.

Grefenstette, E., Amos, B., Yarats, D., Htut, P. M., Molchanov, A., Meier, F., Kiela, D., Cho, K., and Chintala, S. Generalized inner loop meta-learning. *arXiv preprint arXiv:1910.01727*, 2019.

Hazan, E. Introduction to online convex optimization. *arXiv preprint arXiv:1909.05207*, 2019.

Hazan, E., Agarwal, A., and Kale, S. Logarithmic regret algorithms for online convex optimization. *Machine Learning*, 69(2-3):169–192, 2007.

Hazan, E., Singh, K., and Zhang, C. Efficient regret minimization in non-convex games. *arXiv preprint arXiv:1708.00075*, 2017.

Hospedales, T., Antoniou, A., Micaelli, P., and Storkey, A. Meta-learning in neural networks: A survey, 2020.

Krizhevsky, A. et al. Learning multiple layers of features from tiny images. *Technical report*, 2009.

Lake, B. M., Ullman, T. D., Tenenbaum, J. B., and Gershman, S. J. Building machines that learn and think like people. *Behavioral and brain sciences*, 40, 2017.

LeCun, Y., Bottou, L., Bengio, Y., and Haffner, P. Gradient-based learning applied to document recognition. *Proceedings of the IEEE*, 86(11):2278–2324, 1998.

Li, X., Zhou, Y., Wu, T., Socher, R., and Xiong, C. Learn to grow: A continual structure learning framework for overcoming catastrophic forgetting. In Chaudhuri, K. and Salakhutdinov, R. (eds.), *Proceedings of the 36th International Conference on Machine Learning*, volume 97 of *Proceedings of Machine Learning Research*, pp. 3925–3934. PMLR, 09–15 Jun 2019. URL http://proceedings.mlr.press/v97/li19m.html.

Li, Z., Zhou, F., Chen, F., and Li, H. Meta-sgd: Learning to learn quickly for few-shot learning. *arXiv preprint arXiv:1707.09835*, 2017.

Mishra, N., Rohaninejad, M., Chen, X., and Abbeel, P. A simple neural attentive meta-learner. *arXiv preprint arXiv:1707.03141*, 2017.

Nguyen, C. V., Li, Y., Bui, T. D., and Turner, R. E. Variational continual learning. In *International Conference on Learning Representations*, 2018. URL https://openreview.net/forum?id=BkQqq0gRb.

Paszke, A., Gross, S., Massa, F., Lerer, A., Bradbury, J., Chanan, G., Killeen, T., Lin, Z., Gimelshein, N., Antiga, L., Desmaison, A., Kopf, A., Yang, E., DeVito, Z., Raison, M., Tejani, A., Chilamkurthy, S., Steiner, B., Fang, L., Bai, J., and Chintala, S. Pytorch: An imperative style, high-performance deep learning library. In Wallach, H., Larochelle, H., Beygelzimer, A., d'Alché-Buc, F., Fox, E., and Garnett, R. (eds.), *Advances in Neural Information Processing Systems 32*, pp. 8024–8035. Curran Associates, Inc., 2019.

Ravi, S. and Larochelle, H. Optimization as a model for few-shot learning. In *International Conference on Learning Representations*, 2017.

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

Shalev-Shwartz, S. Online learning and online convex optimization. *Foundations and Trends in Machine Learning*, 4, 2012. doi: 10.1561/2200000018.

Shalev-Shwartz, S. and Singer, Y. Online learning: Theory, algorithms, and applications. 2007.

Snell, J., Swersky, K., and Zemel, R. Prototypical networks for few-shot learning. In *Advances in neural information processing systems*, pp. 4077–4087, 2017.

Thrun, S. and Pratt, L. *Learning to learn*. Springer Science & Business Media, 2012.

Vanschoren, J. Meta-learning: A survey. *arXiv preprint arXiv:1810.03548*, 2018.

Vinyals, O., Blundell, C., Lillicrap, T., Wierstra, D., et al. Matching networks for one shot learning. *Advances in neural information processing systems*, 29:3630–3638, 2016.

Yao, H., Zhou, Y., Mahdavi, M., Li, Z. J., Socher, R., and Xiong, C. Online structured meta-learning. In Larochelle, H., Ranzato, M., Hadsell, R., Balcan, M. F., and Lin, H. (eds.), *Advances in Neural Information Processing Systems*, volume 33, pp. 6779–6790. Curran Associates, Inc., 2020. URL https://proceedings.neurips.cc/paper/2020/file/4b86ca48d90bd5f0978afa3a012503a4-Paper.pdf.

Yu, T., Geng, X., Finn, C., and Levine, S. Variable-shot adaptation for online meta-learning. *arXiv preprint arXiv:2012.07769*, 2020.

Zhou, P., Yuan, X., Xu, H., Yan, S., and Feng, J. Efficient meta learning via minibatch proximal update. *Advances in Neural Information Processing Systems*, 32: 1534–1544, 2019.

Zhuang, Z., Wang, Y., Yu, K., and Lu, S. No-regret non-convex online meta-learning. In *ICASSP 2020-2020 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pp. 3942–3946. IEEE, 2020.

Zinkevich, M. Online convex programming and generalized infinitesimal gradient ascent. In *Machine Learning, Proceedings of the Twentieth International Conference (ICML 2003), August 21-24, 2003, Washington, DC, USA*, pp. 928–936. AAAI Press, 2003. URL http://www.aaai.org/Library/ICML/2003/icml03-120.php.