Online Meta-Learning

Chelsea Finn [*1] **Aravind Rajeswaran** [*2] **Sham Kakade** [2] **Sergey Levine** [1]

Abstract

A central capability of intelligent systems is the ability to continuously build upon previous experiences to speed up and enhance learning of new tasks. Two distinct research paradigms have studied this question. Meta-learning views this problem as learning a prior over model parameters that is amenable for fast adaptation on a new task, but typically assumes the tasks are available together as a batch. In contrast, online (regret based) learning considers a setting where tasks are revealed one after the other, but conventionally trains a single model without task-specific adaptation. This work introduces an online meta-learning setting, which merges ideas from both paradigms to better capture the spirit and practice of continual lifelong learning. We propose the follow the meta leader (FTML) algorithm which extends the MAML algorithm to this setting. Theoretically, this work provides an $O(\log T)$ regret guarantee with one additional higher order smoothness assumption (in comparison to the standard online setting). Our experimental evaluation on three different large-scale problems suggest that the proposed algorithm significantly outperforms alternatives based on traditional online learning approaches.

1. Introduction

Two distinct research paradigms have studied how prior tasks or experiences can be used by an agent to inform future learning. Meta-learning (Schmidhuber, 1987; Santoro et al., 2016; Finn et al., 2017) casts this as the problem of *learning to learn*, where past experience is used to acquire a prior over model parameters or a learning procedure, and typically studies a setting where meta-training tasks are made available together upfront. In contrast, online learning (Hannan, 1957; Cesa-Bianchi & Lugosi, 2006) considers a sequential setting where tasks are revealed one after another, but aims to attain zero-shot generalization without any task-specific adaptation. We argue that neither setting is ideal for studying continual lifelong learning. Meta-learning deals with learning to learn, but neglects the sequential and non-stationary aspects of the problem. Online learning offers an appealing theoretical framework, but does not generally consider how past experience can accelerate adaptation to a new task. In this work, we motivate and present the *online meta-learning* problem setting, where the agent simultaneously uses past experiences in a sequential setting to learn good priors, and also adapt quickly to the current task at hand.

As an example, Figure 1 shows a family of sinusoids. Imagine that each task is a regression problem from x to y corresponding to *one* sinusoid. When presented with data from a large collection of such tasks, a naïve approach that does not consider the task structure would collectively use all the data, and learn a prior that corresponds to the model $y = 0$. An algorithm that understands the underlying structure would recognize that each curve in the family is a sinusoid, and would therefore attempt to identify, for a new batch of data, which sinusoid it corresponds to. As another example where joint training fails, Figure 1 also shows colored MNIST digits with different backgrounds. Suppose we've seen MNIST digits with various colored backgrounds, and then observe a "7" on a new color. We might conclude from training on all of the data seen so far that all digits with that color must all be "7." In fact, this is an optimal conclusion from a purely statistical standpoint. However, if we understand that the data is divided into different tasks, and take note of the fact that each task has a different color, a better conclusion is that the color is irrelevant. Training on all of the data together, or only on the new data, does not achieve this goal.

Meta-learning offers an appealing solution: by learning how to learn from past tasks, we can make use of task structure

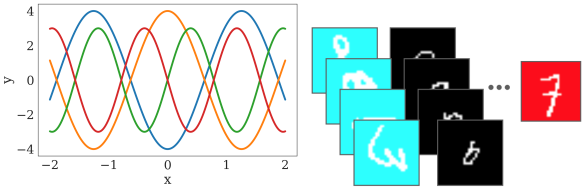

Figure 1. (left) sinusoid functions and (right) colored MNIST

[*]Equal contribution [1]UC Berkeley [2]University of Washington. Correspondence to: Chelsea Finn <cbfinn@stanford.edu>, Aravind Rajeswaran <aravraj@cs.washington.edu>.

Proceedings of the 36^{th} International Conference on Machine Learning, Long Beach, California, PMLR 97, 2019.

and extract information from the data that both allows us to succeed on the current task and adapt to new tasks more quickly. However, typical meta learning approaches assume that a sufficiently large set of tasks are made available upfront for meta-training. In the real world, tasks are likely available only sequentially, as the agent is learning in the world, and also from a non-stationary distribution. By recasting meta-learning in a sequential or online setting, that does not make strong distributional assumptions, we can enable faster learning on new tasks as they are presented.

Our contributions: In this work, we formulate the online meta-learning problem setting and present the *follow the meta-leader (FTML)* algorithm. This extends the MAML algorithm to the online meta-learning setting, and is analogous to follow the leader in online learning. We analyze FTML and show that it enjoys a $O(\log T)$ regret guarantee when competing with the best meta-learner in hindsight. In this endeavor, we also provide the first set of results (under any assumptions) where MAML-like objective functions can be provably and efficiently optimized. We also develop a practical form of FTML that can be used effectively with deep neural networks on large scale tasks, and show that it significantly outperforms prior methods. The experiments involve vision-based sequential learning tasks with the MNIST, CIFAR-100, and PASCAL 3D+ datasets.

2. Foundations

Before introducing online meta-learning, we briefly summarize the foundations of meta-learning, the model-agnostic meta-learning (MAML) algorithm, and online learning. To illustrate the differences in setting and algorithms, we will use the running example of few-shot learning, which we describe below first. We emphasize that online learning, MAML, and the online meta-learning formulations have a broader scope than few-shot supervised learning.

2.1. Few-Shot Learning

In few-shot supervised learning (Santoro et al., 2016), we are interested in a family of tasks, where each task $\mathcal{T}$ is associated with a notional and infinite-size population of input-output pairs. Our goal is to learn a new task $\mathcal{T}_i$ while accessing only a small, finite-size labeled dataset $\mathcal{D}_i := \{\mathbf{x}_i, \mathbf{y}_i\}$ corresponding $\mathcal{T}_i$. If we have a predictive model, $\boldsymbol{h}(\cdot; \mathbf{w})$, with parameters $\mathbf{w}$, the population risk of the model is

$$f_i(\mathbf{w}) := \mathbb{E}_{(\mathbf{x},\mathbf{y})\sim\mathcal{T}_i}[\ell(\mathbf{x}, \mathbf{y}, \mathbf{w})],$$

where the expectation is defined over the task population and ℓ is a loss function, such as the square loss or cross-entropy between the model prediction and the correct label. An example of ℓ corresponding to squared error loss is $\ell(\mathbf{x}, \mathbf{y}, \mathbf{w}) = ||\mathbf{y} - \boldsymbol{h}(\mathbf{x}; \mathbf{w})||^2$.

Let $\mathcal{L}(\mathcal{D}_i, \mathbf{w})$ represent the average loss on the dataset $\mathcal{D}_i$. Effectively minimizing $f_i(\mathbf{w})$ is likely hard if we rely only on $\mathcal{D}_i$ due to the small size of the dataset. However, meta-learning algorithms aim to perform better by drawing upon data from many tasks from the family, as we discuss next.

2.2. Meta-Learning and MAML

Meta-learning, or learning to learn, aims to bootstrap from a set of tasks to learn faster on a new task. Tasks are assumed to be drawn from a fixed distribution, $\mathcal{T} \sim \mathbb{P}(\mathcal{T})$. At meta-training time, M tasks $\{\mathcal{T}_i\}_{i=1}^M$ are drawn from this distribution and datasets corresponding to them are made available to the agent. At deployment time, we are faced with a new test task $\mathcal{T}_j \sim \mathbb{P}(\mathcal{T})$, for which we are again presented with a small dataset $\mathcal{D}_j := \{\mathbf{x}_j, \mathbf{y}_j\}$. Meta-learning algorithms attempt to find a model using the M training tasks, such that when $\mathcal{D}_j$ is revealed from the test task, the model can be quickly updated to minimize $f_j(\mathbf{w})$.

Model-agnostic meta-learning (MAML) (Finn et al., 2017) does so by learning an initial set of parameters $\mathbf{w}_{\text{MAML}}$, such that performing a few steps of gradient descent from $\mathbf{w}_{\text{MAML}}$ using $\mathcal{D}_j$ minimizes $f_j(\cdot)$. To get such an initialization, MAML solves the optimization problem:

$$\mathbf{w}_{\text{MAML}} := \arg\min_{\mathbf{w}} \frac{1}{M}\sum_{i=1}^{M} f_i\big(\mathbf{w} - \alpha\nabla\hat{f}_i(\mathbf{w})\big). \quad (1)$$

The inner gradient $\nabla\hat{f}_i(\mathbf{w})$ is based on a small mini-batch from $\mathcal{D}_i$. Hence, MAML optimizes for few-shot generalization. Finn et al. (2017) show that gradient-based methods can be used to optimize Eq. 1 with existing automatic differentiation libraries. Stochastic gradient methods are used since the population risk is not known. At meta-test time, the solution is fine-tuned as: $\mathbf{w}_j \leftarrow \mathbf{w}_{\text{MAML}} - \alpha\nabla\hat{f}_j(\mathbf{w}_{\text{MAML}})$ with the gradient obtained using $\mathcal{D}_j$. MAML and other meta-learning algorithms (see Section 7) are not directly applicable to sequential settings, as they assume a fixed fixed task distribution and have two distinct-phases, meta-training and meta-testing. We instead would like to develop algorithms that work in continuous learning settings with non-stationary task distributions.

2.3. Online Learning

In online learning, an agent faces a sequence of loss functions $\{f_t\}_{t=1}^{\infty}$, one in each round t. These functions need not be drawn from a fixed distribution, and could even be chosen adversarially over time. The learner must sequentially decide on model parameters $\{\mathbf{w}_t\}_{t=1}^{\infty}$ that perform well on the loss sequence. In particular, the goal is to minimize some notion of regret defined as the difference between the learner's loss, $\sum_{t=1}^{T} f_t(\mathbf{w}_t)$, and the best performance achievable by some family of methods (comparator class). The most standard notion of regret is to compare to the cumulative loss of the best *fixed* model in hindsight:

$$\text{Regret}_T = \sum_{t=1}^{T} f_t(\mathbf{w}_t) - \min_{\mathbf{w}} \sum_{t=1}^{T} f_t(\mathbf{w}). \quad (2)$$

We would like algorithms for which this regret grows with T as slowly as possible. An algorithm whose regret grows sub-linearly in T is non-trivially learning and adapting. One of the simplest algorithms in this setting is follow the leader (FTL) (Hannan, 1957), which computes parameters as:

$$\mathbf{w}_{t+1} = \arg\min_{\mathbf{w}} \sum_{k=1}^{t} f_k(\mathbf{w}).$$

FTL enjoys strong performance guarantees depending on the properties of the loss function. For the few-shot supervised learning example, FTL would consolidate all the data from the prior stream of tasks into a single large dataset and fit a single model to this dataset. As alluded to in Section 1, and as we find in our empirical evaluation in Section 6, such a "joint training" approach may not learn effective models. To overcome this issue, we may desire a more adaptive notion of a comparator class, and algorithms that have low regret against such a comparator, as we will discuss next.

3. The Online Meta-Learning Problem

We consider a sequential setting where an agent is faced with tasks one after another. Each task corresponds to a *round*, denoted by t. In each round, the goal of the learner is to determine model parameters $\mathbf{w}_t$ that perform well for the corresponding task at that round. This is monitored by $f_t : \mathbf{w} \in \mathcal{W} \to \mathbb{R}$, which we would like to be minimized. Crucially, we consider a setting where the agent can perform some local *task-specific* updates to the model before it is deployed and evaluated in each round. This is realized through an update procedure, which at round t, is a mapping $\boldsymbol{U}_t : \mathbf{w} \in \mathcal{W} \to \tilde{\mathbf{w}} \in \mathcal{W}$. This procedure takes as input $\mathbf{w}$ and returns $\tilde{\mathbf{w}}$ that performs better on f_t. One example for $\boldsymbol{U}_t$ is a step of gradient descent (Finn et al., 2017): $\boldsymbol{U}_t(\mathbf{w}) = \mathbf{w} - \alpha \nabla \hat{f}_t(\mathbf{w})$. As specified in Section 2.2, $\nabla \hat{f}_t$ is potentially an approximate gradient of f_t, e.g. obtained using a mini-batch of data from the task at round t. The overall protocol is as follows:

1. At round t, the agent chooses a model defined by $\mathbf{w}_t$.
2. The world simultaneously chooses task defined by f_t.
3. The agent obtains access to the update procedure $\boldsymbol{U}_t$, and uses it to update parameters as $\tilde{\mathbf{w}}_t = \boldsymbol{U}_t(\mathbf{w}_t)$.
4. The agent incurs loss $f_t(\tilde{\mathbf{w}}_t)$. Advance to round $t+1$.

The goal for the agent is to minimize regret over the rounds. A highly ambitious comparator is the best meta-learned model in hindsight. Let $\{\mathbf{w}_t\}_{t=1}^{T}$ be the sequence of models generated by the algorithm. Then, the regret we consider is:

$$\text{Regret}_T = \sum_{t=1}^{T} f_t\big(\boldsymbol{U}_t(\mathbf{w}_t)\big) - \min_{\mathbf{w}} \sum_{t=1}^{T} f_t\big(\boldsymbol{U}_t(\mathbf{w})\big). \quad (3)$$

Notice that we allow the comparator to adapt locally to each task at hand; thus the comparator has strictly more capabilities than the learning agent, since it is presented with all the task functions in batch mode. Here, again, achieving sublinear regret suggests that the agent is improving over time and is competitive with the best meta-learner in hindsight. As discussed earlier, in the batch setting, when faced with multiple tasks, meta-learning performs significantly better than training a single model for all the tasks. Thus, we may hope that learning sequentially, but still being competitive with the best meta-learner in hindsight, provides a significant leap in continual learning.

4. Algorithm and Analysis

In this section, we outline the *follow the meta leader* (FTML) algorithm and provide an analysis of its behavior.

4.1. Follow the Meta Leader

Taking inspiration from the form of the follow the leader algorithm (Hannan, 1957; Kalai & Vempala, 2005), we propose the FTML algorithm template which updates model parameters as:

$$\mathbf{w}_{t+1} = \arg\min_{\mathbf{w}} \sum_{k=1}^{t} f_k\big(\boldsymbol{U}_k(\mathbf{w})\big). \quad (4)$$

This can be interpreted as the agent playing the best meta-learner in hindsight if the learning process were to stop at round t. In the remainder of this section, we will show that under standard assumptions on the losses, and just one additional assumption on higher order smoothness, this algorithm has strong regret guarantees. In practice, we may not have full access to $f_k(\cdot)$, such as when it is the population risk and we only have a finite size dataset. In such cases, we will draw upon stochastic approximation algorithms to solve the optimization problem in Eq. (4).

4.2. Assumptions

We make the following assumptions about each loss function in the learning problem for all t. Let $\boldsymbol{\theta}$ and $\boldsymbol{\phi}$ represent two arbitrary choices of *model parameters*.

Assumption 1. *(C^2-smoothness)*

1. *(Lipschitz in function value) f has gradients bounded by G, i.e. $||\nabla f(\boldsymbol{\theta})|| \leq G \; \forall \; \boldsymbol{\theta}$. This is equivalent to f being $G-$Lipschitz.*
2. *(Lipschitz gradient) f is $\beta-$smooth, i.e. $||\nabla f(\boldsymbol{\theta}) - \nabla f(\boldsymbol{\phi})|| \leq \beta ||\boldsymbol{\theta} - \boldsymbol{\phi}|| \; \forall (\boldsymbol{\theta}, \boldsymbol{\phi})$.*
3. *(Lipschitz Hessian) f has $\rho-$Lipschitz Hessians, i.e. $||\nabla^2 f(\boldsymbol{\theta}) - \nabla^2 f(\boldsymbol{\phi})|| \leq \rho ||\boldsymbol{\theta} - \boldsymbol{\phi}|| \; \forall (\boldsymbol{\theta}, \boldsymbol{\phi})$.*

Assumption 2. *(Strong convexity) Suppose that f is convex. Furthermore, suppose f is $\mu-$strongly convex, i.e. $||\nabla f(\boldsymbol{\theta}) - \nabla f(\boldsymbol{\phi})|| \geq \mu ||\boldsymbol{\theta} - \boldsymbol{\phi}||$.*

These assumptions are largely standard in online learning (Cesa-Bianchi & Lugosi, 2006), except 1.3. Examples where these assumptions hold include logistic regression

and $L2$ regression over a bounded domain. Assumption 1.3 is a statement about the higher order smoothness of functions which is common in non-convex analysis (Nesterov & Polyak, 2006; Jin et al., 2017). In our setting, it allows us to characterize the landscape of the MAML function, which has a gradient step embedded within it. Importantly, these assumptions *do not* trivialize the meta-learning setting. We can observe a clear difference in performance between meta-learning and joint training even when f_i are quadratic functions, which correspond to the simplest strongly convex setting. See Appendix A for an example illustration.

4.3. Analysis

We analyze the FTML algorithm when the update procedure is a single step of gradient descent, as in the formulation of MAML. Concretely, the update procedure we consider is $\boldsymbol{U}_t(\mathbf{w}) = \mathbf{w} - \alpha\nabla\hat{f}_t(\mathbf{w})$. For this update rule, we first state our main theorem below.

Theorem 1. *Suppose f and $\hat{f} : \mathbb{R}^d \to \mathbb{R}$ satisfy assumptions 1 and 2. Let $\tilde{f}$ be the function evaluated after a one step gradient update procedure, i.e.*

$$\tilde{f}(\mathbf{w}) \coloneqq f\big(\mathbf{w} - \alpha\nabla\hat{f}(\mathbf{w})\big).$$

If the step size is selected as $\alpha \leq \min\{\frac{1}{2\beta}, \frac{\mu}{8\rho G}\}$, then $\tilde{f}$ is convex. Furthermore, it is also $\tilde{\beta} = 9\beta/8$ smooth and $\tilde{\mu} = \mu/8$ strongly convex.

For the proof, see Appendix B. The following corollary is now immediate.

Corollary 1. *(inherited convexity for the MAML objective) If $\{f_i, \hat{f}_i\}_{i=1}^K$ satisfy assumptions 1 and 2, then the MAML optimization problem,*

$$\underset{\mathbf{w}}{\text{minimize}}\ \frac{1}{M}\sum_{i=1}^{M} f_i\big(\mathbf{w} - \alpha\nabla\hat{f}_i(\mathbf{w})\big),$$

with $\alpha \leq \min\{\frac{1}{2\beta}, \frac{\mu}{8\rho G}\}$ is convex. Furthermore, it is $9\beta/8$-smooth and $\mu/8$-strongly convex.

Since the objective function is convex, we may expect first-order optimization methods to be effective, since gradients can be efficiently computed with standard automatic differentiation libraries (as discussed in Finn et al. (2017)). In fact, this work provides the first set of results (under any assumptions) under which MAML-like objective function can be provably and efficiently optimized. Another immediate corollary of our main theorem is that FTML now enjoys the same regret guarantees (up to constant factors) as FTL does in the comparable setting (with strongly convex losses).

Corollary 2. *(inherited regret bound for FTML) Suppose that for all t, f_t and $\hat{f}_t$ satisfy assumptions 1 and 2. Suppose that the update procedure in FTML (Eq. 4) is chosen as $\boldsymbol{U}_t(\mathbf{w}) = \mathbf{w} - \alpha\nabla\hat{f}_t(\mathbf{w})$ with $\alpha \leq \min\{\frac{1}{2\beta}, \frac{\mu}{8\rho G}\}$. Then, FTML enjoys the following regret guarantee*

$$\sum_{t=1}^{T} f_t\big(\boldsymbol{U}_t(\mathbf{w}_t)\big) - \min_{\mathbf{w}}\sum_{t=1}^{T} f_t\big(\boldsymbol{U}_t(\mathbf{w})\big) = O\left(\frac{32G^2}{\mu}\log T\right)$$

See Appendix C for a proof. More generally, our main theorem implies that there exists a large family of online meta-learning algorithms that enjoy sub-linear regret, based on the inherited smoothness and strong convexity of $\tilde{f}(\cdot)$. See Hazan (2016); Shalev-Shwartz (2012); Shalev-Shwartz & Kakade (2008) for algorithmic templates to derive sub-linear regret based algorithms.

5. Practical Online Meta-Learning Algorithm

In the previous section, we derived a theoretically principled algorithm for convex losses. Many practical problems have non-convex loss landscales. However, methods developed for convex losses, such as AdaGrad (Duchi et al., 2011), often perform well in non-convex settings. Taking inspiration from these successes, we describe a practical instantiation of FTML, and evaluate its performance in Section 6.

The main considerations when adapting the FTML algorithm to few-shot supervised learning with high capacity neural network models are: (a) the optimization problem in Eq. (4) has no closed form solution, and (b) we do not have access to the population risk f_t but only a subset of the data. To overcome both these limitations, we can use iterative stochastic optimization algorithms. Specifically, by adapting the MAML algorithm (Finn et al., 2017), we can use stochastic gradient descent with a minibatch $\mathcal{D}_k^{\text{tr}}$ as the update rule, and stochastic gradient descent with an independently-sampled minibatch $\mathcal{D}_k^{\text{val}}$ to optimize the parameters. The gradient computation is described below:

$$\begin{aligned} &\mathbf{g}_t(\mathbf{w}) = \nabla_{\mathbf{w}}\,\mathbb{E}_{k\sim\nu^t}\mathcal{L}\big(\mathcal{D}_k^{\text{val}}, \boldsymbol{U}_k(\mathbf{w})\big), \quad \text{where} \\ &\boldsymbol{U}_k(\mathbf{w}) \equiv \mathbf{w} - \alpha\,\nabla_{\mathbf{w}}\,\mathcal{L}\big(\mathcal{D}_k^{\text{tr}}, \mathbf{w}\big) \end{aligned} \tag{5}$$

Here, $\nu^t(\cdot)$ denotes a sampling distribution for the previously seen tasks $\mathcal{T}_1, ..., \mathcal{T}_t$. In our experiments, we use the uniform distribution, $\nu^t \equiv P(k) = 1/t\ \forall k = \{1, 2, \dots t\}$, but other sampling distributions can be used if required. Recall that $\mathcal{L}(\mathcal{D}, \mathbf{w})$ is the loss function (e.g. cross-entropy) averaged over the datapoints $(\mathbf{x}, \mathbf{y}) \in \mathcal{D}$ for the model with parameters $\mathbf{w}$. Using independently sampled minibatches $\mathcal{D}^{\text{tr}}$ and $\mathcal{D}^{\text{val}}$ minimizes interaction between the inner gradient update $\boldsymbol{U}_t$ and the outer optimization (Eq. 4), which is performed using the gradient above ($\mathbf{g}_t$) in conjunction with Adam (Kingma & Ba, 2015). While $\boldsymbol{U}_t$ in Eq. 5 includes only one gradient step, we observed that it is beneficial to take multiple gradient steps in the inner loop (i.e., in $\boldsymbol{U}_t$), which is consistent with prior works (Finn et al., 2017; Grant et al., 2018; Antoniou et al., 2018; Shaban et al., 2018).

Now that we have derived the gradient, the overall algorithm proceeds as follows. We first initialize a task buffer $\mathcal{B} = [\,]$. When presented with a new task at round t, we add task $\mathcal{T}_t$ to $\mathcal{B}$ and initialize a task-specific dataset $\mathcal{D}_t = [\,]$, which is appended to as data incrementally arrives for task $\mathcal{T}_t$. As new data arrives for task $\mathcal{T}_t$, we iteratively compute and

Algorithm 1 Online Meta-Learning with FTML

Input: Performance threshold of proficiency, γ
randomly initialize $\mathbf{w}_1$
initialize the task buffer as empty, $\mathcal{B} \leftarrow [\,]$
for $t = 1, \ldots$ **do**
  initialize $\mathcal{D}_t = \emptyset$
  Add $\mathcal{B} \leftarrow \mathcal{B} + [\,\mathcal{T}_t\,]$
  while $|\mathcal{D}_{\mathcal{T}_t}| < N$ **do**
    Append batch of n new datapoints $\{(\mathbf{x}, \mathbf{y})\}$ to $\mathcal{D}_t$
    $\mathbf{w}_t \leftarrow$ `Meta-Update`$(\mathbf{w}_t, \mathcal{B}, t)$
    $\tilde{\mathbf{w}}_t \leftarrow$ `Update-Procedure` $(\mathbf{w}_t, \mathcal{D}_t)$
    if $\mathcal{L}\left(\mathcal{D}_t^{\text{test}}, \tilde{\mathbf{w}}_t\right) < \gamma$ **then**
      Record efficiency for task $\mathcal{T}_t$ as $|\mathcal{D}_t|$ datapoints
    end if
  end while
  Record final performance of $\tilde{\mathbf{w}}_t$ on test set $\mathcal{D}_t^{\text{test}}$ for task t.
  $\mathbf{w}_{t+1} \leftarrow \mathbf{w}_t$
end for

Algorithm 2 FTML Subroutines

Input: Hyperparameters parameters α, η
function `Meta-Update`$(\mathbf{w}, \mathcal{B}, t)$
  for $n_{\text{m}} = 1, \ldots, N_{\text{meta}}$ steps **do**
    Sample task $\mathcal{T}_k$: $k \sim \nu^t(\cdot)$ *// (or a minibatch of tasks)*
    Sample minibatches $\mathcal{D}_k^{\text{tr}}$, $\mathcal{D}_k^{\text{val}}$ uniformly from $\mathcal{D}_k$
    Compute gradient $\mathbf{g}_t$ using $\mathcal{D}_k^{\text{tr}}$, $\mathcal{D}_k^{\text{val}}$, and Eq. 5
    Update parameters $\mathbf{w} \leftarrow \mathbf{w} - \eta\, \mathbf{g}_t$ *// (or use Adam)*
  end for
  Return $\mathbf{w}$
end function
function `Update-Procedure`$(\mathbf{w}, \mathcal{D})$
  Initialize $\tilde{\mathbf{w}} \leftarrow \mathbf{w}$
  for $n_{\text{g}} = 1, \ldots, N_{\text{grad}}$ steps **do**
    $\tilde{\mathbf{w}} \leftarrow \tilde{\mathbf{w}} - \alpha \nabla \mathcal{L}(\mathcal{D}, \tilde{\mathbf{w}})$
  end for
  Return $\tilde{\mathbf{w}}$
end function

apply the gradient in Eq. 5, which uses data from all tasks seen so far. Once all of the data (finite-size) has arrived for $\mathcal{T}_t$, we move on to task $\mathcal{T}_{t+1}$. This procedure is detailed in Alg. 1, including the evaluation, which we discuss next.

To evaluate performance of the model at any point within a particular round t, we first update the model as using all of the data ($\mathcal{D}_t$) seen so far within round t. This is outlined in the `Update-Procedure` subroutine of Algorithm 2. Note that this is different from the update $\mathbf{U}_t$ used within the meta-optimization, which uses a fixed-size minibatch since many-shot meta-learning is computationally expensive and memory intensive. In practice, we meta-train with update minibatches of size at-most 25, whereas evaluation may use hundreds of datapoints for some tasks. After the model is updated, we measure the performance using held-out data $\mathcal{D}_t^{\text{test}}$ from task $\mathcal{T}_t$. This data is not revealed to the online meta-learner at any time. Further, we also evaluate task learning efficiency, which corresponds to the size of $\mathcal{D}_t$ required to achieve a specified performance threshold γ, e.g. $\gamma = 90\%$ classification accuracy or γ corresponds to a certain loss value. If less data is sufficient to reach the threshold, then priors learned from previous tasks are being useful and we have achieved positive transfer.

6. Experimental Evaluation

Our experimental evaluation studies the practical FTML algorithm (Section 5) in the context of vision-based online learning problems. These problems include synthetic modifications of the MNIST dataset, pose detection with synthetic images based on PASCAL3D+ models (Xiang et al., 2014), and online classification with the CIFAR-100 dataset. The aim of our experimental evaluation is to study the following questions: (1) can online meta-learning (specifically FTML) be successfully applied to multiple non-stationary learning problems? and (2) does online meta-learning (FTML) provide empirical benefits over prior methods? To this end, we compare to the following: (a) Train on everything (*TOE*) jointly trains a single predictve model on all available data so far (including $\mathcal{D}_t$ at round t). (b) Train *from scratch* initializes $\mathbf{w}_t$ randomly and trains using $\mathcal{D}_t$. (c) Joint training with fine-tuning, which at round t, trains jointly until round $t-1$, and then finetunes it specifically to round t using only $\mathcal{D}_t$. This corresponds to the standard *FTL* approach, followed by task-specific fine-tuning.

We note that TOE is a very strong point of comparison, capable of reusing representations across tasks, as has been proposed in a number of prior continual learning works (Rusu et al., 2016; Aljundi et al., 2017; Wang et al., 2017). However, unlike FTML, TOE does not explicitly learn the structure across tasks. Thus, it may not be able to fully utilize the information present in the data, and hence may not be able to learn new tasks with only a few examples. Further, the model might incur negative transfer if the new task differs substantially from previous ones, as has been observed in prior work (Parisotto et al., 2016). Training on each task from scratch avoids negative transfer, but also precludes any reuse between tasks. When the amount of data for a given task is large, we may expect training from scratch to perform well since it can avoid negative transfer and can learn specifically for the particular task. Finally, FTL with fine-tuning represents a natural online learning comparison, which in principle should combine the best parts of learning from scratch and TOE, since this approach adapts specifically to each task *and* benefits from prior data. However, in contrast to FTML, this method does not explicitly meta-learn and hence may not fully utilize any structure in the tasks.

6.1. Rainbow MNIST

In this experiment, we create a sequence of tasks based on the MNIST dataset. We transform the digits in a num-

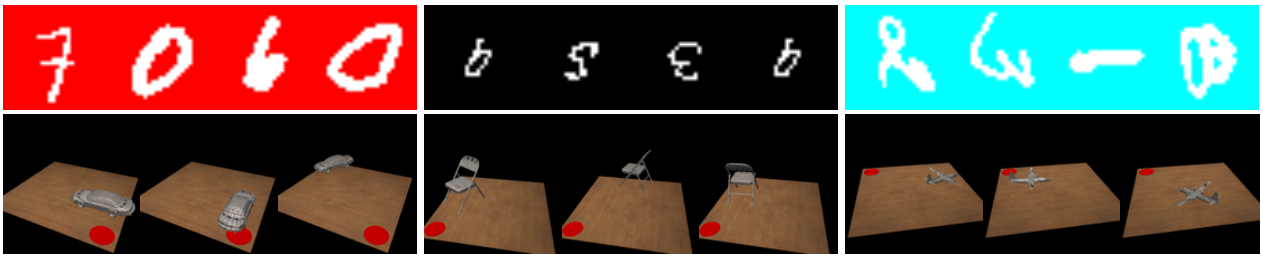

Figure 2. Illustration of three tasks for Rainbow MNIST (top) and pose prediction (bottom). CIFAR images not shown. Rainbow MNIST includes different rotations, scales, and background colors. For pose prediction, the goal is to predict the global pose of the object on the table. Cross-task variation includes 50 object models within 9 object classes, varying object scales, and different camera viewpoints.

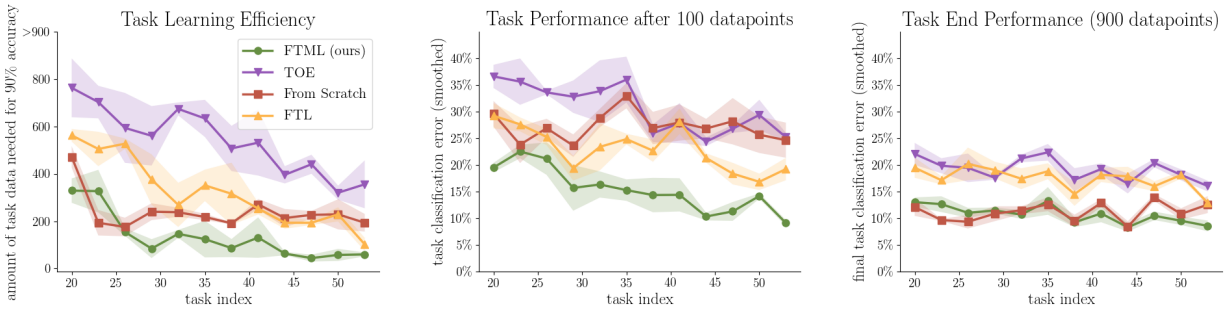

Figure 3. Rainbow MNIST results. Left: amount of data needed to learn each new task. Center: task performance after 100 datapoints on the current task. Right: The task performance after all 900 datapoints for the current task have been received. Lower is better for all plots, while shaded regions show standard error over three random seeds. FTML can learn new tasks more and more efficiently as each new task is received, demonstrating effective forward transfer.

ber of ways to create different tasks, including 7 colored backgrounds, 2 scales (half size and original size), and 4 rotations of 90 degree intervals. As illustrated in Fig. 2, a task involves correctly classifying digits with a randomly sampled background, scale, and rotation. This leads to 56 total tasks. We partitioned the MNIST training dataset into 56 batches of examples, each with 900 images and applied the corresponding task transformation to each batch of images. The ordering of tasks was selected at random and we set 90% classification accuracy as the proficiency threshold.

Learning curves in Fig. 3 show that FTML learns tasks more and more quickly, with each new task added. We also find that FTML substantially outperforms the alternative approaches in both efficiency and final performance. FTL performs better than TOE since it uses task-specific adaptation, but its performance is still inferior to FTML. We hypothesize that, while the prior methods improve in efficiency over the course of learning as they see more tasks, they struggle to prevent negative transfer on each new task. Lastly, training from scratch does not learn efficiently compared to models that incorporate data from other tasks; but, their final performance with 900 datapoints is similar.

6.2. Five-Way CIFAR-100

In this experiment, we create a sequence of 5-way classification tasks based on the CIFAR-100 dataset, which contains more challenging and realistic RGB images than MNIST. Each classification problem involves a newly-introduced class from the 100 classes in CIFAR-100. Thus, different tasks correspond to different labels spaces. The ordering of tasks is selected at random, and we measure performance using classification accuracy. Since it is less clear what the proficiency threshold should be for this task, we evaluate the accuracy on each task after varying numbers of datapoints have been seen. Since these tasks are mutually exclusive (as label space is changing), it makes sense to train the TOE model with a different final layer for each task. An extremely similar approach to this is to use our meta-learning approach but to only allow the final layer parameters to be adapted to each task. Further, such a meta-learning approach is a more direct comparison to our full FTML method, and the comparison can provide insight into whether online meta-learning is simply learning features and performing training on the last layer, or if it is adapting the features to each task. Thus, we compare to this last layer online meta-learning approach instead of TOE with multiple heads. The results (see Figure 4) indicate that FTML learns more efficiently than independent models and a model with a shared feature space. The results on the right indicate that training from scratch achieves good performance with 2000 datapoints, reaching similar performance to FTML. However, the last layer variant of FTML seems to not have the capacity to reach good performance on all tasks.

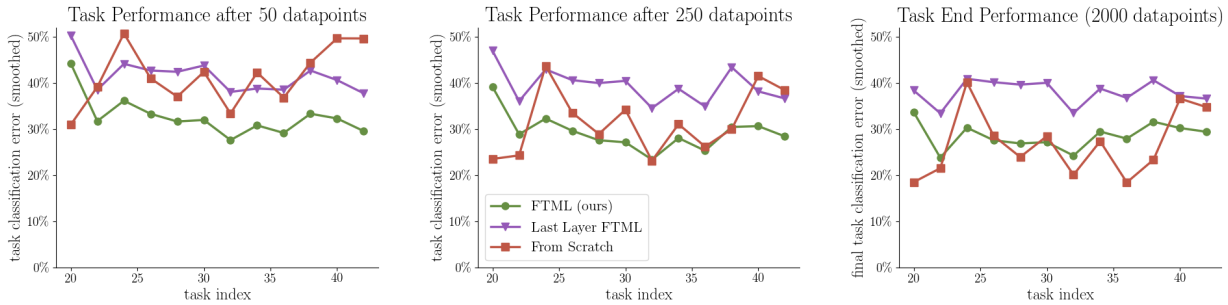

Figure 4. Online CIFAR-100 results, evaluating task performance after seeing 50, 250, and 2000 datapoints for each task. FTML learns each task much more efficiently than models trained from scratch, while both achieve similar asymptotic performance after 2000 datapoints. FTML benefits from adapting all layers rather than learning a shared feature space across tasks while adapting only the last layer.

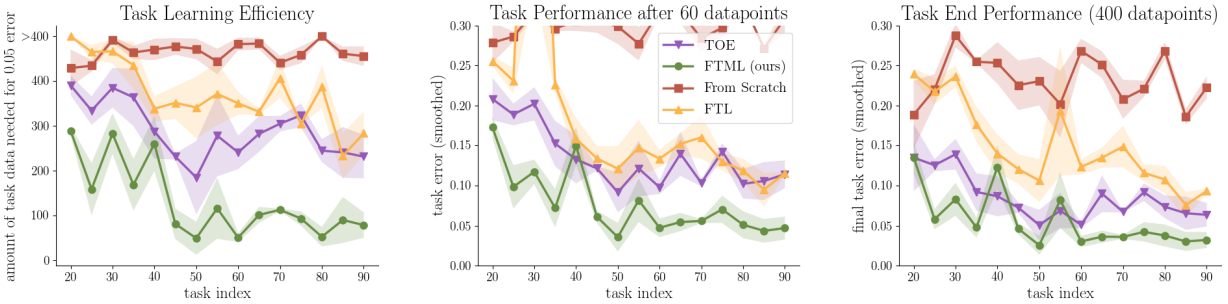

Figure 5. Object pose prediction results. Left: we observe that online meta-learning leads to faster learning as more and more tasks are introduced, learning with only tens of datapoints for many of the tasks. Center & right, we see that meta-learning enables transfer not just for faster learning but also for more effective performance when 60 and 400 datapoints of each task are available. The task order is randomized, leading to spikes when more difficult tasks are introduced. Shaded regions show standard error across three random seeds

6.3. Sequential Object Pose Prediction

In our final experiment, we study a 3D pose prediction problem. Each task involves learning to predict the global position and orientation of an object in an image. We construct a dataset of synthetic images using 50 object models from 9 different object classes in the PASCAL3D+ dataset (Xiang et al., 2014), rendering the objects on a table using the renderer accompanying the MuJoCo physics engine (Todorov et al., 2012) (see Figure 2). To place an object on the table, we select a random 2D location, as well as a random azimuthal angle. Each task corresponds to a different object with a randomly sampled camera angle. We place a red dot on one corner of the table to provide a global reference point for the position. Using this setup, we construct 90 tasks (with an average of about 2 camera viewpoints per object), with 1000 datapoints per task. All models are trained to regress to the global 2D position and the sine and cosine of the azimuthal angle (the angle of rotation along the z-axis). For the loss functions, we use mean-squared error, and set the proficiency threshold to an error of 0.05. We show the results of this experiment in Figure 5. The results demonstrate that meta-learning can improve both efficiency and performance of new tasks over the course of learning, solving many of the tasks with only 10 datapoints. Unlike the previous settings, TOE substantially outperforms training from scratch, indicating that it can effectively make use of the previous data from other tasks, likely due to the greater structural similarity between the pose detection tasks. However, the performance of FTML suggests that even better transfer can be accomplished by explicitly optimizing for the ability to quickly and effectively learn new tasks. Finally, we find that FTL performs comparably or worse than TOE, indicating that task-specific fine-tuning can lead to overfitting when the model is not explicitly trained for the ability to fine-tune effectively.

7. Connections to Related Work

Meta-learning: Our work proposes to use meta-learning or learning to learn (Thrun & Pratt, 1998; Schmidhuber, 1987; Naik & Mammone, 1992), in the context of online (regret-based) learning. Prior works have proposed learning update rules, selective copying of weights, or optimizers (Hochreiter et al., 2001; Bengio et al., 1992; Andrychowicz et al., 2016; Li & Malik, 2017; Ravi & Larochelle, 2017; Schmidhuber, 2002), as well as recurrent models that learn by ingesting datasets directly (Santoro et al., 2016; Duan et al., 2016; Wang et al., 2016; Munkhdalai & Yu, 2017; Mishra et al., 2017). Some meta-learning works have considered online learning at *meta-test time* (Santoro et al., 2016; Al-Shedivat et al., 2017; Nagabandi et al., 2018). However, with the exception of work on online hyperparameter adaptation (Elfwing et al., 2017; Meier et al., 2017; Baydin et al.,

2017), nearly all prior meta-learning algorithms assume that the *meta-training tasks* come from a stationary distribution, and does not consider tasks that are presented as a continuous stream. In contrast, we consider a more flexible approach that allows for adapting all of the model's parameters during online meta-training. Recent work has also considered meta-training with non-stationary distributions using Dirichlet process mixture models over parameters (Grant et al., 2019). In contrast, we introduce a simple extension onto the MAML algorithm without mixtures over parameters, and provide theoretical guarantees.

Continual learning: Our problem setting is related to continual, or lifelong learning (Thrun, 1998; Zhao & Schmidhuber, 1996). A number of papers in this area have focused on avoiding forgetting or negative backward transfer (Goodfellow et al., 2013; Kirkpatrick et al., 2017; Zenke et al., 2017; Rebuffi et al., 2017; Shin et al., 2017; Shmelkov et al., 2017; Lopez-Paz et al., 2017; Nguyen et al., 2017; Schmidhuber, 2013), and maintaining a small model capacity as new tasks are added (Lee et al., 2017; Mallya & Lazebnik, 2017). In this paper, we sidestep the problem of catastrophic forgetting by maintaining a buffer of all the observed data (Isele & Cosgun, 2018), and instead focus on maximizing the efficiency of learning new tasks within a non-stationary meta-learning setting. Furthermore, unlike prior works (Ruvolo & Eaton, 2013; Rusu et al., 2016; Aljundi et al., 2017; Wang et al., 2017), we focus on the setting where there are several tens or hundreds of tasks, and therefore more information that can be transferred from previous tasks to enable few-shot acquisition of new concepts.

Online learning: Similar to continual learning, online learning considers a sequential setting with streaming tasks. Much of the prior work in this area has focused on computationally cheap algorithms that do not iterate over past data multiple times (Cesa-Bianchi & Lugosi, 2006; Hazan et al., 2006; Zinkevich, 2003; Shalev-Shwartz, 2012). Again, we sidestep computational considerations to first study the meta-learning analog of FTL. For this, we derived the FTML algorithm which has low regret when compared to a powerful adaptive comparator class, and demonstrated empirical gains over strong baselines.

Adaptive notions of regret have been considered in prior work to overcome limitations of a fixed comparator. In the dynamic regret setting (Herbster & Warmuth, 1995; Yang et al., 2016; Besbes et al., 2015), the online learner is compared with the sequence of optimal solutions corresponding to each loss function. Unfortunately, lower bounds (Yang et al., 2016) suggest that the comparator class is too powerful and may not provide for any non-trivial learning in the general case. To overcome this challenge, prior work has placed restrictions on how quickly the loss functions or comparator model can change (Hazan & Comandur, 2009; Hall & Willett, 2015; Herbster & Warmuth, 1995). In contrast, we develop a new notion of adaptive regret where the learner and comparator both have access to an update procedure. The update procedures allow the comparator to produce different models for different loss functions, thereby serving as a powerful comparator class (in comparison to a fixed model in hindsight). In this setting, we derived sublinear regret algorithms without placing restrictions on the loss functions. Concurrent work has also studied algorithms related to first order variants of MAML using theoretical tools from online learning (Alquier et al., 2016; Denevi et al., 2019; Khodak et al., 2019). These works also derive regret and generalization bounds, but the algorithms have not yet been empirically studied in large scale domains or non-stationary settings. We believe that our online meta-learning setting captures the spirit and practice of continual lifelong learning, and also shows promising empirical results.

8. Discussion and Future Work

We introduced the online meta-learning setting, which provides a natural perspective on the ideal real-world learning procedure: an intelligent agent interacting with a constantly changing environment should utilize streaming experience to both master the task at hand, and become more proficient at learning new tasks in the future. We proposed and analyzed the FTML algorithm to derive regret bounds, and illustrated how FTML can be adapted to a practical algorithm. Our experiments demonstrate that FTML outperforms prior methods, learing new tasks more and more efficiently over time. We next outline avenues for future work.

More powerful update procedures. We analyzed the case where the update $\boldsymbol{U}_t$ is one gradient step. However, in practice, MAML is often used with multiple gradient steps. Analyzing this case, and potentially higher order update rules, will make for exciting future work.

Memory and computational constraints. In this work, we primarily aimed to discern if it is possible to meta-learn in a sequential setting. As discussed in Section 7, the cost of FTL (and FTML) grows over time as new tasks and loss functions are accumulated. Further, in many practical online learning problems, it is challenging to store all datapoints from previous tasks. While we showed that our method can effectively learn nearly 100 tasks in sequence without significant burdens on compute or memory, scalability remains a concern. Can a more streaming algorithm like mirror descent that does not store all the past experiences be successful as well? Our main theoretical results (Section 4.3) suggests that there exist a large family of online meta-learning algorithms that enjoy sublinear regret. Tapping into the large body of work in online learning, particularly mirror descent, to develop computationally cheaper algorithms would make for exciting future work.

Acknowledgements

Aravind Rajeswaran thanks Emo Todorov for valuable discussions on the problem formulation. This work was supported by the National Science Foundation via IIS-1651843, Google, Amazon, and NVIDIA. Sham Kakade acknowledges funding from the Washington Research Foundation Fund for Innovation in Data-Intensive Discovery and the NSF CCF 1740551 award.

References

Al-Shedivat, M., Bansal, T., Burda, Y., Sutskever, I., Mordatch, I., and Abbeel, P. Continuous adaptation via meta-learning in nonstationary and competitive environments. *CoRR*, abs/1710.03641, 2017.

Aljundi, R., Chakravarty, P., and Tuytelaars, T. Expert gate: Lifelong learning with a network of experts. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2017.

Alquier, P., Mai, T. T., and Pontil, M. Regret bounds for lifelong learning. In *AISTATS*, 2016.

Andrychowicz, M., Denil, M., Gomez, S., Hoffman, M. W., Pfau, D., Schaul, T., and de Freitas, N. Learning to learn by gradient descent by gradient descent. In *Neural Information Processing Systems (NIPS)*, 2016.

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

Baydin, A. G., Cornish, R., Rubio, D. M., Schmidt, M., and Wood, F. Online learning rate adaptation with hypergradient descent. *arXiv:1703.04782*, 2017.

Bengio, S., Bengio, Y., Cloutier, J., and Gecsei, J. On the optimization of a synaptic learning rule. In *Optimality in Artificial and Biological Neural Networks*, 1992.

Besbes, O., Gur, Y., and Zeevi, A. J. Non-stationary stochastic optimization. *Operations Research*, 63:1227–1244, 2015.

Cesa-Bianchi, N. and Lugosi, G. Prediction, learning, and games. 2006.

Denevi, G., Ciliberto, C., Grazzi, R., and Pontil, M. Learning-to-learn stochastic gradient descent with biased regularization. *CoRR*, abs/1903.10399, 2019.

Duan, Y., Schulman, J., Chen, X., Bartlett, P. L., Sutskever, I., and Abbeel, P. Rl2: Fast reinforcement learning via slow reinforcement learning. *arXiv:1611.02779*, 2016.

Duchi, J., Hazan, E., and Singer, Y. Adaptive subgradient methods for online learning and stochastic optimization. *Journal of Machine Learning Research*, 2011.

Elfwing, S., Uchibe, E., and Doya, K. Online meta-learning by parallel algorithm competition. *arXiv:1702.07490*, 2017.

Finn, C., Abbeel, P., and Levine, S. Model-agnostic meta-learning for fast adaptation of deep networks. *International Conference on Machine Learning (ICML)*, 2017.

Goodfellow, I. J., Mirza, M., Xiao, D., Courville, A., and Bengio, Y. An empirical investigation of catastrophic forgetting in gradient-based neural networks. *arXiv:1312.6211*, 2013.

Grant, E., Finn, C., Levine, S., Darrell, T., and Griffiths, T. Recasting gradient-based meta-learning as hierarchical bayes. *International Conference on Learning Representations (ICLR)*, 2018.

Grant, E., Jerfel, G., Heller, K., and Griffiths, T. L. Modulating transfer between tasks in gradient-based meta-learning, 2019.

Hall, E. C. and Willett, R. M. Online convex optimization in dynamic environments. *IEEE Journal of Selected Topics in Signal Processing*, 9:647–662, 2015.

Hannan, J. Approximation to bayes risk in repeated play. *Contributions to the Theory of Games*, 1957.

Hazan, E. Introduction to online convex optimization. 2016.

Hazan, E. and Comandur, S. Efficient learning algorithms for changing environments. In *ICML*, 2009.

Hazan, E., Kalai, A. T., Kale, S., and Agarwal, A. Logarithmic regret algorithms for online convex optimization. *Machine Learning*, 69:169–192, 2006.

Herbster, M. and Warmuth, M. K. Tracking the best expert. *Machine Learning*, 32:151–178, 1995.

Hochreiter, S., Younger, A. S., and Conwell, P. R. Learning to learn using gradient descent. In *International Conference on Artificial Neural Networks*, 2001.

Isele, D. and Cosgun, A. Selective experience replay for lifelong learning. *arXiv preprint arXiv:1802.10269*, 2018.

Jin, C., Ge, R., Netrapalli, P., Kakade, S. M., and Jordan, M. I. How to escape saddle points efficiently. In *ICML*, 2017.

Kalai, A. T. and Vempala, S. Efficient algorithms for online decision problems. *J. Comput. Syst. Sci.*, 71:291–307, 2005.

Khodak, M., Balcan, M.-F., and Talwalkar, A. S. Provable guarantees for gradient-based meta-learning. *CoRR*, abs/1902.10644, 2019.

Kingma, D. and Ba, J. Adam: A method for stochastic optimization. *International Conference on Learning Representations (ICLR)*, 2015.

Kirkpatrick, J., Pascanu, R., Rabinowitz, N., Veness, J., Desjardins, G., Rusu, A. A., Milan, K., Quan, J., Ramalho, T., Grabska-Barwinska, A., et al. Overcoming catastrophic forgetting in neural networks. *Proceedings of the National Academy of Sciences*, 2017.

Lee, J., Yun, J., Hwang, S., and Yang, E. Lifelong learning with dynamically expandable networks. *arXiv:1708.01547*, 2017.

Levine, S., Finn, C., Darrell, T., and Abbeel, P. End-to-end training of deep visuomotor policies. *The Journal of Machine Learning Research*, 2016.

Li, K. and Malik, J. Learning to optimize. *International Conference on Learning Representations (ICLR)*, 2017.

Lopez-Paz, D. et al. Gradient episodic memory for continual learning. In *Advances in Neural Information Processing Systems*, 2017.

Mallya, A. and Lazebnik, S. Packnet: Adding multiple tasks to a single network by iterative pruning. *arXiv:1711.05769*, 2017.

Meier, F., Kappler, D., and Schaal, S. Online learning of a memory for learning rates. *arXiv:1709.06709*, 2017.

Mishra, N., Rohaninejad, M., Chen, X., and Abbeel, P. Meta-learning with temporal convolutions. *arXiv preprint arXiv:1707.03141*, 2017.

Munkhdalai, T. and Yu, H. Meta networks. *International Conference on Machine Learning (ICML)*, 2017.

Nagabandi, A., Finn, C., and Levine, S. Deep online learning via meta-learning: Continual adaptation for model-based rl. *arXiv preprint arXiv:1812.07671*, 2018.

Naik, D. K. and Mammone, R. Meta-neural networks that learn by learning. In *International Joint Conference on Neural Netowrks (IJCNN)*, 1992.

Nesterov, Y. and Polyak, B. T. Cubic regularization of newton method and its global performance. *Math. Program.*, 108:177–205, 2006.

Nguyen, C. V., Li, Y., Bui, T. D., and Turner, R. E. Variational continual learning. *arXiv:1710.10628*, 2017.

Parisotto, E., Ba, J. L., and Salakhutdinov, R. Actor-mimic: Deep multitask and transfer reinforcement learning. *International Conference on Learning Representations (ICLR)*, 2016.

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

Rebuffi, S.-A., Kolesnikov, A., and Lampert, C. H. icarl: Incremental classifier and representation learning. In *Proc. CVPR*, 2017.

Rusu, A. A., Rabinowitz, N. C., Desjardins, G., Soyer, H., Kirkpatrick, J., Kavukcuoglu, K., Pascanu, R., and Hadsell, R. Progressive neural networks. *arXiv:1606.04671*, 2016.

Ruvolo, P. and Eaton, E. Ella: An efficient lifelong learning algorithm. In *International Conference on Machine Learning*, pp. 507–515, 2013.

Santoro, A., Bartunov, S., Botvinick, M., Wierstra, D., and Lillicrap, T. Meta-learning with memory-augmented neural networks. In *International Conference on Machine Learning (ICML)*, 2016.

Schmidhuber, J. Evolutionary principles in self-referential learning. *Diploma thesis, Institut f. Informatik, Tech. Univ. Munich*, 1987.

Schmidhuber, J. Optimal ordered problem solver. *Machine Learning*, 54:211–254, 2002.

Schmidhuber, J. Powerplay: Training an increasingly general problem solver by continually searching for the simplest still unsolvable problem. In *Front. Psychol.*, 2013.

Shaban, A., Cheng, C.-A., Hirschey, O., and Boots, B. Truncated back-propagation for bilevel optimization. *CoRR*, abs/1810.10667, 2018.

Shalev-Shwartz, S. Online learning and online convex optimization. *"Foundations and Trends in Machine Learning"*, 2012.

Shalev-Shwartz, S. and Kakade, S. M. Mind the duality gap: Logarithmic regret algorithms for online optimization. In *NIPS*, 2008.

Shin, H., Lee, J. K., Kim, J., and Kim, J. Continual learning with deep generative replay. In *Advances in Neural Information Processing Systems*, 2017.

Shmelkov, K., Schmid, C., and Alahari, K. Incremental learning of object detectors without catastrophic forgetting. *arXiv:1708.06977*, 2017.

Singh, A., Yang, L., and Levine, S. Gplac: Generalizing vision-based robotic skills using weakly labeled images. *arXiv:1708.02313*, 2017.

Szegedy, C., Vanhoucke, V., Ioffe, S., Shlens, J., and Wojna, Z. Rethinking the inception architecture for computer vision. In *Conference on Computer Vision and Pattern Recognition (CVPR)*, 2016.

Thrun, S. Lifelong learning algorithms. In *Learning to learn*. Springer, 1998.

Thrun, S. and Pratt, L. *Learning to learn*. Springer Science & Business Media, 1998.

Todorov, E., Erez, T., and Tassa, Y. Mujoco: A physics engine for model-based control. In *International Conference on Intelligent Robots and Systems (IROS)*, 2012.

Wang, J. X., Kurth-Nelson, Z., Tirumala, D., Soyer, H., Leibo, J. Z., Munos, R., Blundell, C., Kumaran, D., and Botvinick, M. Learning to reinforcement learn. *arXiv:1611.05763*, 2016.

Wang, Y.-X., Ramanan, D., and Hebert, M. Growing a brain: Fine-tuning by increasing model capacity. In *IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR)*, 2017.

Xiang, Y., Mottaghi, R., and Savarese, S. Beyond pascal: A benchmark for 3d object detection in the wild. In *Conference on Applications of Computer Vision (WACV)*, 2014.

Yang, T., Zhang, L., Jin, R., and Yi, J. Tracking slowly moving clairvoyant: Optimal dynamic regret of online learning with true and noisy gradient. In *ICML*, 2016.

Zenke, F., Poole, B., and Ganguli, S. Continual learning through synaptic intelligence. In *International Conference on Machine Learning*, 2017.

Zhao, J. and Schmidhuber, J. Incremental self-improvement for life-time multi-agent reinforcement learning. In *From Animals to Animats 4: International Conference on Simulation of Adaptive Behavior*, 1996.

Zinkevich, M. Online convex programming and generalized infinitesimal gradient ascent. In *ICML*, 2003.