How Fine-Tuning Allows for Effective Meta-Learning

Anonymous Author(s)
Affiliation
Address
email

Abstract

Representation learning has served as a key tool for meta-learning, enabling rapid learning of new tasks. Recent works like MAML learn task-specific representations by finding an initial representation requiring minimal per-task adaptation (*i.e.* a fine-tuning-based objective). We present a theoretical framework for analyzing a MAML-like algorithm, assuming all available tasks require *approximately* the same representation. We then provide risk bounds on predictors found by fine-tuning via gradient descent, demonstrating that the method provably leverages the shared structure. We illustrate these bounds in the logistic regression and neural network settings. In contrast, we establish settings where learning one representation for all tasks (*i.e.* using a "frozen representation" objective) fails. Notably, *any such algorithm* cannot outperform directly learning the target task with no other information, in the worst case. This separation underscores the benefit of fine-tuning-based over "frozen representation" objectives in few-shot learning.

1 Introduction

Meta-learning (Thrun & Pratt, 2012) has emerged as an essential tool for adapting prior knowledge to new tasks under data and computational constraints. In this context, a meta-learner has access to related source tasks from a shared environment. The learner aims to uncover inductive biases from the source tasks to reduce the sample/computational complexity of learning new tasks from the same environment. A common approach is representation learning (Bengio et al., 2013), *i.e.* learning a feature extractor from the source tasks. At test time, a learner adapts to a new task by fine-tuning the representation and retraining the final layer(s) (see, *e.g.*, prototype networks (Snell et al., 2017)). Substantial improvements over directly learning a single task has been shown in few-shot learning (Antoniou et al., 2018), a natural setting in many applications including reinforcement learning (Mendonca et al., 2019; Finn et al., 2017), computer vision (Nichol et al., 2018), federated learning (McMahan et al., 2017) and robotics (Al-Shedivat et al., 2017).

The empirical success of representation learning has led to an increased interest in theoretical analyses of underlying phenomena. Recent work assumes an explicitly shared representation across tasks (Du et al., 2020; Tripuraneni et al., 2020b,a; Saunshi et al., 2020; Balcan et al., 2015). For instance, Du et al. (2020) prove a generalization bound consisting of a representation error term and an estimation error term. However, without fine-tuning the whole network, the representation error accumulates to the target task and is *irreducible*, even with an infinitely large target task dataset. This result is consistent with empirical findings, which suggest that fine-tuning the whole network provides substantial performance gains, compared to just learning the final linear layer (Chen et al., 2020; Salman et al., 2020). Due to the lack of (representation) fine-tuning while training on the source tasks, we refer to methods above as making use of "frozen representation" objectives.

Linear separability of tasks on the same features poses additional problems, being an unrealistic assumption when transferring knowledge to other domains (*e.g.* from ImageNet to medical images (Raghu et al., 2019)). Therefore, we consider a more realistic setting where the available tasks only

Submitted to 35th Conference on Neural Information Processing Systems (NeurIPS 2021). Do not distribute.

approximately share the same representation. In this setting, we propose a theoretical framework for analyzing the sample complexity of fine-tuning, using an initial representation from a MAML-like algorithm. We show that fine-tuning quickly adapts to new tasks, requiring fewer samples in certain cases compared to methods using "frozen representation" objectives. To the best of our knowledge, no prior studies exist beyond fine-tuning a linear model (Denevi et al., 2018; Konobeev et al., 2020; Collins et al., 2020a; Lee et al., 2020) or only the task-specific layers (Du et al., 2020; Tripuraneni et al., 2020b,a; Mu et al., 2020). Tripuraneni et al. (2020b), in particular, acknowledge that their work does not incorporate representation fine-tuning, leaving such analysis to future work. It is this gap which we now seek to address.

The following outlines this paper and its contributions:

In **Section 2**, we outline the general setting and our overall assumptions. Additionally, we introduce ADAPTREP, the representative fine-tuning-based algorithm we analyze in this work. As a baseline, we also formally define FROZENREP, which makes use of a "frozen representation" objective.

In **Section 3**, we provide an in-depth analysis of the (d-dimensional) linear representation setting, when learning a k-dimensional representation. First, we show that ADAPTREP achieves rates of

$$r_{\text{source}} = O\left(\frac{kd}{n_{\text{S}}T} + \frac{k}{n_{\text{S}}} + \delta_0\sqrt{\frac{\operatorname{tr}\Sigma}{n_{\text{S}}}}\right) \quad \text{and} \quad r_{\text{target}} = O\left(\frac{k}{n_{\text{T}}} + \delta_0\sqrt{\frac{\operatorname{tr}\Sigma}{n_{\text{T}}}} + r_{source}\right)$$

on the source and target tasks, respectively, when the norm of the representation change is at most δ_0. Thus, fine-tuning can handle the misspecified setting. In contrast, FROZENREP has a minimax rate of $\Omega(d/n_{\text{T}})$ on the target task under certain task distributions, matching standard linear regression. We provide a formal construction and an experimental verification of the gap in **Section B**.

In **Section 4**, we extend the analysis to general function classes. We provide risk bounds of the form

$$\varepsilon_{\text{OPT}} + \varepsilon_{\text{EST}} + \varepsilon_{\text{REPR}},$$

where $\varepsilon_{\text{OPT}}, \varepsilon_{\text{EST}}, \varepsilon_{\text{REPR}}$ are the optimization, estimation, and representation errors, respectively.

The *optimization error* ε_{OPT} quantifies the error from using approximate minima found during optimization. To control ε_{OPT}, our analysis accounts for nonconvexity introduced by representation fine-tuning, and is presented as a self-contained result in **Section G**.

The *estimation error* ε_{EST} stems from fine-tuning to the target task with finite n_{T} samples. It therefore scales with $1/\sqrt{n_{\text{T}}}$ and is controlled by the complexity of the target fine-tuning set.

The *representation error* $\varepsilon_{\text{REPR}}$ is the error incurred by representation learning on the source tasks while adapting to them. It consists of two terms: one scaling as $1/\sqrt{n_{\text{S}}T}$ for learning an initialization using all T source tasks, and the other scaling as $1/\sqrt{n_{\text{S}}}$ for learning task-specific adaptations.

In **Section 5**, we instantiate our guarantees in the two-layer neural network setting. Additionally, we provide an analysis for logistic regression in **Section E**. Furthermore, we extend the linear hard case to a nonlinear setting in **Section F**.

1.1 Related Work

The empirical success of MAML (Finn et al., 2017), and meta-learning in general, has elicited further theoretical analysis from both statistical and optimization perspectives. A flurry of work engages in developing more efficient and theoretically-sound optimization algorithms (Antoniou et al., 2018; Nichol et al., 2018; Li et al., 2017) or convergence analyses (Fallah et al., 2020; Zhou et al., 2019; Rajeswaran et al., 2019; Collins et al., 2020b). Inspired by MAML, a subseqent line of gradient-based meta-learning algorithms have been developed and widely used (Nichol et al., 2018; Al-Shedivat et al., 2017; Jerfel et al., 2018). Much follow-up work focused on the online setting, with accompanying regret bounds (Denevi et al., 2018; Finn et al., 2019; Khodak et al., 2019; Balcan et al., 2015; Alquier et al., 2017; Bullins et al., 2019; Pentina & Lampert, 2014).

The statistical analysis of meta-learning traces back to Baxter (2000); Maurer & Jaakkola (2005), which focused on inductive bias learning. Following the same setting, Amit & Meir (2018); Konobeev et al. (2020); Maurer et al. (2016); Pentina & Lampert (2014) fix a meta-distribution for source task sampling, and measure the average generalization error/gap over the meta distribution. Another line of work connects the target performance to source using distance measures between distributions (Ben-David & Borbely, 2008; Ben-David et al., 2010; Mohri & Medina, 2012). Finally,

another series of works studied the benefits of using additional "side information" provided with a task for specializing the parameters of an inner algorithm (Denevi et al., 2020, 2021).

The hardness of meta-learning has also been investigated under various settings. Recent work has studied meta-learning performance in worst-case settings (Collins et al., 2020b; Hanneke & Kpotufe, 2020a,b; Kpotufe & Martinet, 2018; Lucas et al., 2020). Hanneke & Kpotufe (2020a) provide a no-free-lunch result with a problem-independent minimax lower bound, while Konobeev et al. (2020) also provide problem-dependent lower bound on a simple linear setting.

2 General Setting

2.1 Notation

Let $[n] \coloneqq \{1, \ldots, n\}$. We denote the vector L_2-norm as $\|\cdot\|_2$, and the matrix Frobenius norm as $\|\cdot\|_F$. Additionally, $\langle \cdot, \cdot \rangle$ can denote either the Euclidean inner product or the Frobenius inner product between matrices.

For a matrix A, we let $\sigma_i(A)$ denote its ith largest singular value. Additionally, for positive semidefinite A, we write $\lambda_{\max}(A)$ and $\lambda_{\min}(A)$ for its largest and smallest eigenvalues, and $A^{1/2}$ for its principal square root. We write P_A for the projection onto the column span of A, denoted $\operatorname{Col} A$, and $P_A^{\perp} \coloneqq I - P_A$ for the projection onto its complement.

We use standard O, Θ, and Ω notation to denote orders of growth. We also use $a \lesssim b$ or $a \ll b$ to indicate that $a = O(b)$. Finally, we write $a \asymp b$ for $a = \Theta(b)$.

2.2 Problem Setting

Preliminaries. Fix an input space $\mathcal{X} \subseteq \mathbb{R}^d$ and label space $\mathcal{Y} \subseteq \mathbb{R}$. We assume that the learner has access to T source tasks. Each task $t \in [T]$ is associated with a distribution μ_t over the set $\mathcal{X} \times \mathcal{Y}$ of input-label pairs. From each μ_t, the learner observes n_{S} i.i.d. samples $\{(x_{i,t}, y_{i,t}) \mid i \in [n_{\mathrm{S}}]\}$, which we collect into a matrix $X_t \in \mathbb{R}^{n_{\mathrm{S}} \times d}$ and a vector $y_t \in \mathbb{R}^{n_{\mathrm{S}}}$ for convenience. Finally, we measure learner performance using the loss function $\ell : \mathbb{R} \times \mathcal{Y} \to \mathbb{R}$.

We aim to find common structure within the source tasks that could be leveraged for future tasks. In this work, such common structure is defined using *representations*, mappings $\phi : \mathcal{X} \to \mathcal{Z}$ from the input space to a latent space $\mathcal{Z} \subseteq \mathbb{R}^k$. These representations lie in some function class parameterized by some normed space. However, unlike prior work, we *do not* assume that all tasks use a single representation. Instead, there is a fixed ϕ_{θ_0}, such that the representation ϕ_{θ_t} associated with any task t is near ϕ_{θ_0}, *i.e.* $\|\theta_t - \theta_0\|$ is small. Task t then has predictor $x \mapsto \langle w_t, \phi_{\theta_t}(x) \rangle$ for some $w_t \in \mathbb{R}^k$.

AdaptRep Procedure. Following the discussion above, consider the objective

$$\min_{\theta_0} \min_{\substack{\theta_t, w_t \\ \|\theta_t - \theta_0\| \le \delta_0}} \frac{1}{n_{\mathrm{S}} T} \sum_{t=1}^{T} \sum_{i=1}^{n_{\mathrm{S}}} \ell(\langle w_t, \phi_{\theta_t}(x_{i,t}) \rangle, y_{i,t}) \tag{1}$$

for some fixed $\delta_0 > 0$, which will be the main focus of this work. We refer to (1) as AdaptRep, since it finds an initialization ϕ_{θ_0} for which every task has a good representation nearby (and thus only a small fine-tuning step is needed for any task). (1) can be viewed as a constrained form of algorithms found in the literature such as iMAML (Rajeswaran et al., 2019) and Meta-MinibatchProx (Zhou et al., 2019). However, we do not use train-validation splits, as is widespread in practice. This setup is motivated by results in Bai et al. (2020), which show that data splitting may be undesirable, assuming realizability. Furthermore, empirical evaluations have demonstrated successes despite the lack of such splits (Zhou et al., 2019).

FrozenRep Procedure. As a baseline for comparisons, we will also be studying the ubiquitous "frozen representation" objective, which is analyzed in detail in Du et al. (2020); Tripuraneni et al. (2020b). Formally, this objective can be written as the following optimization problem:

$$\min_{\theta_0} \min_{w_t} \frac{1}{n_{\mathrm{S}} T} \sum_{t=1}^{T} \sum_{i=1}^{n_{\mathrm{S}}} \ell(\langle w_t, \phi_{\theta_0}(x_{i,t}) \rangle, y_{i,t}). \tag{2}$$

Since ϕ_{θ_0} is frozen for all source tasks, we refer to the method above as FROZENREP. In Sections 3.4 and F, we will prove that unlike ADAPTREP, there are cases where FROZENREP is unable to leverage the latent structure in the presence of misspecifications.

Learning Target Tasks. Let θ_0 be a solution to (1) or (2). The learner obtains n_{T} i.i.d. samples $\{(x_i, y_i)\}$ from a new task with distribution μ, which we collect into $X \in \mathbb{R}^{n_{\mathrm{T}} \times d}$ and $y \in \mathbb{R}^{n_{\mathrm{T}}}$ as before. We adapt to the target task using θ_0 by solving

$$\min_{\substack{\theta, w \\ \|\theta - \theta_0\| \leq \delta}} \frac{1}{n_{\mathrm{T}}} \sum_{i=1}^{n_{\mathrm{T}}} \ell(\langle w, \phi_\theta(x_i) \rangle, y_i). \tag{3}$$

for some $\delta \geq 0$. This learned predictor is evaluated via its population loss over μ, defined as

$$\mathcal{L}_\infty^\mu(\theta, w) \coloneqq \mathbb{E}_{(x,y)\sim\mu} \left[\ell(\langle w, \phi_\theta(x) \rangle, y) \right].$$

We focus on the few-shot learning setting for the target task, where n_{T} is small, and a learner needs to effectively use the source tasks to sufficiently learn the target task.

Statistical Assumptions and Optimization Oracles. For ADAPTREP to be sensible, we need to ensure that a desirable initialization exists. Accordingly, we assume that for some θ_0^*, every task t is associated with (θ_t^*, w_t^*) where $\|\theta_t^* - \theta_0^*\| \leq \delta_0$, so that the associated distribution μ_t is given by

$$x \sim p, \quad y \mid x \sim q(\cdot \mid \left\langle w_t^*, \phi_{\theta_t^*}(x) \right\rangle).$$

Here, p is an input distribution (same for all tasks), while q models label noise. Specifically, in regression, we set $\mathbb{E}[y \mid x] = \left\langle w_t^*, \phi_{\theta_t^*}(x) \right\rangle$. With an appropriate choice of loss function ℓ, we can guarantee that the optimal predictor under the population loss is $x \mapsto \left\langle w_t^*, \phi_{\theta_t^*}(x) \right\rangle$.

Throughout the paper, we assume access to an oracle for solving (1) or (2), similar to Du et al. (2020); Tripuraneni et al. (2020b). For detailed analyses of source training optimization, we refer the reader to Ji et al. (2020); Wang et al. (2020). On the other hand, representation fine-tuning introduces nonconvexity during target time not present in prior work, where one simply optimizes a final linear layer (a convex problem). Thus, our bounds explicitly incorporate optimization performance on (3). We focus on using projected gradient descent (PGD), which applies to a wide variety of settings, under certain loss landscape assumptions. These standalone results are also provided in Section G.

3 ADAPTREP in the Linear Setting

We first examine ADAPTREP in the linear setting to illustrate key ideas. Here, representations are linear transformations $\mathbb{R}^d \to \mathbb{R}^k$ for $d > k$ so that $\phi_B(x) = B^\top x$, with distances measured by the Frobenius norm. In this setting, we provide a performance bound for ADAPTREP, and then exhibit a specific instance for which FROZENREP fails, establishing our claimed sample complexity gap.

3.1 Statistical Assumptions

Data Sampling. We specialize the statistical assumptions of Section 2.2. Assume that the input distribution p is zero-mean and has covariance Σ. Let $\kappa = \lambda_{\max}(\Sigma)/\lambda_{\min}(\Sigma)$ be the condition number of Σ. As in Du et al. (2020), we impose the following tail condition on p:

Assumption 3.1. *There exists $\rho > 0$ such that if $x \sim p$, then $\Sigma^{-1/2}x$ is ρ^2-sub-Gaussian*[1].

This assumption is used in the proofs to guarantee probabilistic tail bounds, and can be replaced with other suitable tail conditions. Finally, we define $q(\cdot \mid \mu) \sim \mathcal{N}(\mu, \sigma^2)$ for a fixed $\sigma > 0$.

Parameterizing the Tasks. Let B^* be the initial representation, and Δ_t^* be the task-specific change for task t, where $\|\Delta_t^*\|_F \leq \delta_0$. Furthermore, let $w_t^* \in \mathbb{R}^k$ be the task-specific weights. The predictor for task t is then given by $x \mapsto \langle x, (B^* + \Delta_t^*) w_t^* \rangle$. We define $\theta_t^* = (B^* + \Delta_t^*) w_t^*$, and combine (w_t^*) into a matrix $W^* = [w_1^*, \ldots, w_T^*] \in \mathbb{R}^{k \times T}$.

Assumption 3.2 (Source task diversity)**.** *For any $t \in [T]$, $\|w_t^*\|_2 = \Theta(1)$, and $\sigma_k^2(W^*) = \Omega(T/k)$.*

[1] A zero-mean random vector v is ρ^2-sub-Gaussian if for any fixed u, $\mathbb{E}\left[\exp(\lambda v^\top u)\right] \leq \lambda^2 \|u\|_2^2 \rho^2/2$.

Since $\sum_{i\in[k]} \sigma_i^2(W^*) = \|W^*\|_F^2 = \Theta(T)$ by Assumption 3.2, the bound on $\sigma_k^2(W^*)$ implies that the weights covers all directions in $\mathbb{R}^k$ roughly evenly. This condition is satisfied with high probability when the w_t^* are sampled from a sub-Gaussian distribution with well-conditioned covariance.

To visualize the parameters, note that $\theta_t^* = B^* w_t^* + \delta_t^*$, where $\|\delta_t^*\|_2 = \|\Delta_t^* w_t^*\|_2 = O(\delta_0)$ by Assumption 3.2. Thus, the predictor weights lie *near the rank-k subspace determined by B^**. As a convention since the parameterization is not unique, we require $(B^*)^\top \Sigma \delta_t^* = 0$ for any $t \in [T]$. This is without loss of generality, as we can always redefine relevant quantities appropriately.

Finally, we evaluate the learner on a target task $\theta^* := B^* w^* + \delta^*$ for some w^* and $\|\delta^*\|_2 \le \delta_0$.

3.2 Training Procedure

Source training. We consider the following regularized form of (1):

$$\min_B \min_{\Delta_t, w_t} \frac{1}{2n_{\mathrm{S}}T} \sum_{t=1}^{T} \|y_t - X_t(B + \Delta_t) w_t\|_2^2 + \frac{\lambda}{2} \|\Delta_t\|_F^2 + \frac{\gamma}{2} \|w_t\|_2^2 . \tag{4}$$

As will be shown in Section A, the regularization is equivalent to regularizing $\sqrt{\lambda\gamma}\,\|\Delta_t w_t\|_2$, which is consistent with the intuition that $\Delta_t^* w_t^*$ has small norm. This regularization is necessary, since (1) only controls the norm of Δ_t, which is insufficient for controlling $\Delta_t w_t$.

Target training. Letting B_0 be an orthonormalized solution to (4), we adapt to the target task via

$$\mathcal{L}_\beta(\Delta, w) = \frac{1}{2n} \|y - \beta X \left(A_{B_0} + \Delta\right)(w_0 + w)\|_2^2 , \tag{5}$$

where $A_{B_0} := [B_0 \; B_0] \in \mathbb{R}^{d\times 2k}$ and $w_0 = [u, -u]$ for a fixed unit-norm vector $u \in \mathbb{R}^k$. This corresponds to training a predictor of the form $x \mapsto \langle x, (A_{B_0} + \Delta)(w_0 + w)\rangle$.

To optimize the objective, we perform T_{PGD} steps of PGD with stepsize η on (5) with

$$\mathcal{C}_\beta := \{(\Delta, w) \mid \|\Delta\|_F \le c_1/\beta, \|w\|_2 \le c_2/\beta\}$$

as the feasible set, where we explicitly define c_1 and c_2 in Section A.

To interpret $\mathcal{L}_\beta$ and $\mathcal{C}_\beta$, note that the linear predictor has parameter $\theta = A_{B_0} w + \Delta w_0 + \Delta w$, since $A_{B_0} w_0 = 0$. Within $\mathcal{C}_\beta$, the first two terms are of norm $O(1)$, while Δw has norm $O(1/\beta)$. Thus, for large β, the cross term Δw is a negligible perturbation. Consequently, the predictor is approximately linear in (Δ, w), and thus $\mathcal{L}_\beta$ is approximately convex in (Δ, w), when $(\Delta, w) \in \mathcal{C}_\beta$.

3.3 Performance Bound

Now, we provide a performance bound on the performance of the algorithm proposed in Section 3.2. We define the following rates[2] of interest, where $R_\Sigma(w^*, \delta_0) > 0$:

$$r_{\mathrm{S}}(n_{\mathrm{S}}, T) := \frac{\sigma^2 kd}{n_{\mathrm{S}}T} + \sigma\delta_0 \|\Sigma\|_2^{1/2} \sqrt{\frac{kd}{n_{\mathrm{S}}T}} + \frac{\sigma^2 k}{n_{\mathrm{S}}} + \frac{\sigma\delta_0}{\sqrt{n_{\mathrm{S}}}} \sqrt{\operatorname{tr}\Sigma}$$

$$r_{\mathrm{T}}(n_{\mathrm{T}}) := \frac{\sigma^2 k}{n_{\mathrm{T}}} + \frac{\sigma\delta_0}{\sqrt{n_{\mathrm{T}}}} \sqrt{\operatorname{tr}\Sigma} + \min\left[k r_{\mathrm{S}}(n_{\mathrm{S}}, T) \;,\; R_\Sigma(w^*, \delta_0) \sqrt{k r_{\mathrm{S}}(n_{\mathrm{S}}, T)} \frac{\sigma}{\sqrt{n_{\mathrm{T}}}} \sqrt{\operatorname{tr}\Sigma} \right]$$

Theorem 3.1 (Performance guarantee, linear representations)**.** *Assume that Assumptions 3.1 and 3.2 hold, $n_{\mathrm{S}} \gg \rho^4(d + \log(T/\delta))$, and $n_{\mathrm{T}} \gg \rho^4(d + \log(1/\delta))$. Then there are $(\lambda, \gamma, \beta, T_{\mathrm{PGD}}, \eta, c_1, c_2)$ (specified in Section A) such that the training procedure in Section 3.2, with high probability, finds θ achieving excess risk bounded as*

$$\mathbb{E}\left[(x^\top \theta^* - x^\top \theta)^2\right] \lesssim \min\left(r_{\mathrm{T}}(n_{\mathrm{T}}),\; \frac{\sigma^2 d}{n_{\mathrm{T}}} \right).$$

We prove this result in Section A. Nevertheless, we give an interpretation of r_{S} and r_{T} below.

[2] Log factors and non-dominant terms suppressed for clarity. Full rates are presented in the appendix.

Source Training. The rate r_{S} represents the performance of the learner during source training. Despite misspecifications, the algorithm is able to pool the aggregate $n_{\mathrm{S}}T$ samples to learn a common (kd-dimensional) initialization, as shown by the first two terms. The last two terms represent the rate achieved on each task *upon inferring the initialization*, and therefore decays only with n_{S}. The first is the cost of learning w_t^*, while the second is the cost of learning δ_t^*.

Target Training. The rate r_{T} represents the performance of the learner *with fine-tuning*. The first two terms and their interpretations are analogous to that of the last two terms in r_{S}. On the other hand, the minimum indicates two ways to handle the representation error from source training:

Ignore source error. In the most data-starved regime, we ignore the finite-sample source error, restricting fine-tuning to only learn δ^*. Thus, we incur an irreducible error in the form of $kr_{\mathrm{S}}(n_{\mathrm{S}}, T)$[3].

Fix source error. Alternatively, with more data, we can use fine-tuning to *fix the source representation error*. This additional complexity is captured by the second argument, indicating fine-tuning to a vector with norm $R_\Sigma(w^*, \delta_0)\sqrt{kr_{\mathrm{S}}(n_{\mathrm{S}}, T)}$. Note that the additional norm shrinks to 0 as $n_{\mathrm{S}} \to \infty$.

Finally, by ignoring the representation B_0 completely, we can obtain the alternative trivial $\sigma^2 d/n_{\mathrm{T}}$ rate, which matches the minimax lower bound for standard linear regression (see e.g., Duchi (2016)).

3.4 A Hard Case for FROZENREP

In this section, we present a sample complexity separation between ADAPTREP and FROZENREP in the linear setting. *We provide the formal construction and an experimental verification of the gap in Section B.* Nevertheless, we provide the following informal result:

Theorem 3.2 (FROZENREP-ADAPTREP Sample Complexity Separation)**.** *Assume $k = \Theta(1) \ll d$. There exists a family of task distributions $\mathcal{T}$ satisfying the conditions of Section 3.1, such that with high probability over the draw of $n_{\mathrm{T}} \gtrsim d$ target samples, we have the minimax bound*

$$\min_{\hat{w},\hat{\delta}} \max_{\substack{\tau\in\mathcal{T} \\ \theta^*\in\mathrm{supp}\ \tau}} \mathbb{E}\left[\frac{1}{n_{\mathrm{T}}}\left\|X(\theta^* - \bar{B}\hat{w} - \hat{\delta})\right\|_2^2\right] \gtrsim \frac{\sigma^2 d}{n_{\mathrm{T}}}.$$

FROZENREP *is provided with infinitely many source tasks and per-task samples from τ to learn $\bar{B}$, and n_{T} samples from θ^* to learn $(\hat{w}, \hat{\delta})$. In contrast, by specializing Theorem 3.1 for the same family,* ADAPTREP *(with only finitely many source tasks and per-task samples) achieves excess risk bounded as*

$$\mathbb{E}\left[(x^\top\theta^* - x^\top\hat{\theta})^2\right] \lesssim \min\left(\frac{\sigma}{\sqrt{n_{\mathrm{T}}}}\ ,\ \frac{\sigma^2 d}{n_{\mathrm{T}}}\right).$$

Interpreting the minimax result. Observe that the rate is achievable by performing standard linear regression directly on the target samples, with no other information. Thus, FROZENREP fails to capture the shared structure from source tasks, in the worst case. Furthermore, in the high-dimensional setting when $n_{\mathrm{T}} \asymp d$, there exists a strict separation between ADAPTREP and FROZENREP that widens as $n_{\mathrm{T}} \to \infty$.

The minimax bound applies regardless of the target fine-tuning procedure in use, including those used in practice, *e.g.* iMAML, MetaOptNet (Lee et al., 2019), and R2D2 (Bertinetto et al., 2019). As FROZENREP is provided with infinitely many source tasks and samples, this failure is thus entirely due to the representation learning algorithm. Altogether, we have established a case where incorporating representation fine-tuning is *provably sufficient* for handling misspecifications, unlike ubiquitous "frozen representation" objectives.

4 ADAPTREP in the Nonlinear Setting

We now describe a framework for analyzing fine-tuning in general function classes. To simplify notation, we modify the setting in Section 2.2 so that both the representation ϕ and predictor weights w are captured by a parameter $\theta \in \Theta$, with corresponding predictor g_θ.

Loss Notation. We denote the population loss of predictor h with respect to g as

$$\mathcal{L}_\infty^g(h) := \mathbb{E}_{(x,y)\sim\mu_g}\left[\ell(h(x), y)\right],$$

[3]For *average* target performance, $kr_{\mathrm{S}}(n_{\mathrm{S}}, T)$ can be replaced by $r_{\mathrm{S}}(n_{\mathrm{S}}, T)$; see Section E; Du et al. (2020).

where μ_g samples $x \sim p$ and $y \mid x \sim q(\cdot|g(x))$. Additionally, we let $\mathcal{L}^g(h)$ denote the corresponding finite-sample quantity[4].

Feasible Predictors. For a fixed initial parameter θ and a fine-tuning set $\mathcal{C} \subseteq \Theta$, we define the set

$$\mathcal{A}_\theta^{\mathcal{C}} \coloneqq \{g_{\theta'} \mid \theta' - \theta \in \mathcal{C}\}.$$

Intuitively, this set contains all feasible predictors given an initialization θ and feasible fine-tuning set $\mathcal{C}$. For convenience, we also define the set

$$\left(\mathcal{A}_\theta^{\mathcal{C}}\right)^{\otimes T} \coloneqq \left\{(x_t)_{t\in[T]} \mapsto (f_t(x_t))_{t\in[T]} \;\middle|\; f_1, \ldots, f_T \in \mathcal{A}_\theta^{\mathcal{C}}\right\},$$

which is a collection of functions mapping $\mathcal{X}^T \to \mathcal{Y}^T$. This set represents all feasible ways to choose a predictor for each of the T source tasks, for a given initialization θ and fine-tuning set $\mathcal{C}$.

4.1 Training Procedure

Source Training. Fix a set of possible initializations $\Theta_0 \subseteq \Theta$, and a fine-tuning set $\mathcal{C}_\mathrm{S}$. With n_S samples for each task (with optimal predictor g_t^*), we consider the objective

$$\theta_0 \in \operatorname*{argmin}_{\theta\in\Theta_0} \frac{1}{T}\sum_{t=1}^{T} \min_{g_t \in \mathcal{A}_\theta^{\mathcal{C}_\mathrm{S}}} \mathcal{L}^{g_t^*}(g_t). \tag{6}$$

Note that this corresponds to the AdaptRep objective in (1) using the combined parameter θ.

Target Training. Fix a fine-tuning set $\mathcal{C}_\mathrm{T}$ (not necessarily equal to $\mathcal{C}_\mathrm{S}$). Given n_T samples from a new task (with optimal predictor g^*), we run PGD on $L(\delta) = \mathcal{L}^{g^*}(g_{\theta_0+\delta})$ with feasible set $\mathcal{C}_\mathrm{T}$. During optimization, PGD is run with T_PGD timesteps and step size η.

4.2 Assumptions

We impose the following standard regularity conditions on the loss function ℓ:

Assumption 4.1. *For any $y \in \mathcal{Y}$, $\ell(\cdot, y)$ is 1-Lipschitz[5] and convex, and $|\ell(0, y)| \le B$.*

Task Diversity. To ensure transfer from source to target, we impose the following condition, a specific instance of which was proposed by Du et al. (2020) in the linear setting, and proposed by Tripuraneni et al. (2020b) for general settings:

Assumption 4.2 (Source tasks are (ν, ε)-diverse)**.** *There exists constants (ν, ε) such that if ρ is the distribution of target tasks, then for any $\theta \in \Theta_0$,*

$$\mathbb{E}_{g^*\sim\rho}\left[\inf_{g\in\mathcal{A}_\theta^{\mathcal{C}_\mathrm{T}}} \mathcal{L}_\infty^{g^*}(g) - \mathcal{L}_\infty^{g^*}(g^*)\right] \le \frac{1}{\nu}\left[\frac{1}{T}\sum_{t=1}^{T} \inf_{g\in\mathcal{A}_\theta^{\mathcal{C}_\mathrm{S}}} \mathcal{L}_\infty^{g_t^*}(g) - \mathcal{L}_\infty^{g_t^*}(g_t^*)\right] + \varepsilon.$$

The (ν, ε)-diversity assumption connects the average source task performance (right) to the average target performance (left). Thus, it ensures that optimizing (6) results in controlled target performance. Note that we weaken the condition in Tripuraneni et al. (2020b) to bound average target performance rather than worst-case, as is more suitable for higher-dimensional settings.

Target Loss Landscape. Finally, we impose conditions ensuring that PGD finds a near-optimal solution in $\mathcal{C}_\mathrm{T}$. We remark that these assumptions are specific to our choice of fine-tuning algorithm.

Assumption 4.3 (Approximate linearity in fine-tuning)**.** *Let $x_1, \ldots, x_{n_\mathrm{T}}$ be the set of target inputs. Then, there exists β, L such that*

$$\sup_{\delta\in\mathcal{C}_T} \frac{1}{n_\mathrm{T}}\sum_{i=1}^{n_\mathrm{T}} \left\|\nabla_\theta^2 g_{\theta+\delta}(x_i)\right\|_2^2 \le \beta^2 \quad \textit{and} \quad \sup_{\theta\in\Theta_0} \frac{1}{n_\mathrm{T}}\sum_{i=1}^{n_\mathrm{T}} \left\|\nabla_\theta g_\theta(x_i)\right\|_2^2 \le L^2.$$

Assumption 4.4. $\sup_{\delta\in\mathcal{C}_\mathrm{T}} \|\delta\|_2 \le R$ *for some R.*

[4] Note that we have omitted the samples from the notation for brevity.

[5] This is not restrictive as one can simply rescale the loss, and is assumed for simplicity of presentation.

4.3 Performance Bound

Recall that for a set $\mathcal{H}$ of functions $\mathbb{R}^d \to \mathbb{R}^k$ on n samples, its Rademacher complexity $\mathcal{R}_n(\mathcal{H})$ on n samples is given by

$$\mathcal{R}_n(\mathcal{H}) \coloneqq \mathbb{E}_{\varepsilon,X}\left[\frac{1}{n}\left|\sup_{h\in\mathcal{H}}\sum_{i=1}^{n}\sum_{j=1}^{k}\varepsilon_{ij}h_j(x_i)\right|\right],$$

where (ε_{ij}) are i.i.d. Rademacher random variables and (x_i) are i.i.d. samples from some (preset) distribution. We now proceed with our performance guarantee.

Theorem 4.1 (General Performance Bound)**.** *Assume that all assumptions in Section 4.2 hold. Let (θ_t) be the set of iterates generated by PGD following the procedure in Section 4.1 (step size η specified in Section C). Then, with probability at least $1-\delta$ over the random draw of samples,*

$$\mathbb{E}_{g^*\sim\rho}\left[\min_t \mathcal{L}_\infty^{g^*}(\theta_t) - \mathcal{L}_\infty^{g^*}(\theta^*+\delta^*)\right] \lesssim \underbrace{\beta R^2 + R\sqrt{\frac{L^2+\beta^2R^2}{T_{\mathrm{PGD}}}}}_{\varepsilon_{\mathrm{OPT}}} + \underbrace{\frac{1}{\delta}\mathcal{R}_{n_{\mathrm{T}}}\left(\mathcal{A}_{\theta_0}^{\mathcal{C}_{\mathrm{T}}}\right) + \frac{B}{\delta\sqrt{n_{\mathrm{T}}}}}_{\varepsilon_{\mathrm{EST}}}$$
$$+ \underbrace{\frac{1}{\nu}\left\{\frac{1}{\delta T}\mathcal{R}_{n_{\mathrm{S}}}\left[\bigcup_{\theta\in\Theta_0}\left(\mathcal{A}_\theta^{\mathcal{C}_{\mathrm{S}}}\right)^{\otimes T}\right] + \frac{B}{\delta\sqrt{n_{\mathrm{S}}T}}\right\}}_{\varepsilon_{\mathrm{REPR}}} + \varepsilon\,.$$

Note that the $\mathcal{R}_{n_{\mathrm{T}}}$ complexity term samples from p. Meanwhile, the $\mathcal{R}_{n_{\mathrm{S}}}$ complexity term samples from $p^{\otimes T}$, which concatenates T i.i.d. samples from p (one for each task) for every draw.

We prove Theorem 4.1 in Section C. The Rademacher complexity terms decay in most settings as

$$\frac{1}{T}\mathcal{R}_{n_{\mathrm{S}}}\left[\bigcup_{\theta\in\Theta_0}\left(\mathcal{A}_\theta^{\mathcal{C}_{\mathrm{S}}}\right)^{\otimes T}\right] = O\left(\frac{C(\Theta_0)}{\sqrt{n_{\mathrm{S}}\mathrm{T}}} + \frac{\operatorname{diam}\mathcal{C}_{\mathrm{S}}}{\sqrt{n_{\mathrm{S}}}}\right) \quad \text{and} \quad \mathcal{R}_{n_{\mathrm{T}}}(\mathcal{A}_{\theta_0}^{\mathcal{C}_{\mathrm{T}}}) = O\left(\frac{\operatorname{diam}\mathcal{C}_{\mathrm{T}}}{\sqrt{n_{\mathrm{T}}}}\right),$$

where $C(\Theta_0)$ measures the complexity of Θ_0, and $\operatorname{diam}\mathcal{C}_{\mathrm{S}}$ and $\operatorname{diam}\mathcal{C}_{\mathrm{T}}$ measures the size of the fine-tuning sets $\mathcal{C}_{\mathrm{S}}$ and $\mathcal{C}_{\mathrm{T}}$.

Understanding the Bound. We briefly outline out how the assumptions in Section 4.2 contribute to the final bound. Let θ_{OPT} denote the best solution found by PGD, and define the parameters

$$\theta_{\mathrm{ERM}} \coloneqq \operatorname*{argmin}_{\theta\in\mathcal{A}_{\theta_0}^{\mathcal{C}_{\mathrm{T}}}} \mathcal{L}^{g^*}(g_\theta) \quad \text{and} \quad \bar{\theta} \coloneqq \operatorname*{argmin}_{\theta\in\mathcal{A}_{\theta_0}^{\mathcal{C}_{\mathrm{T}}}} \mathcal{L}_\infty^{g^*}(g_\theta).$$

That is, θ_{ERM} is the minimizer of the empirical risk while $\bar{\theta}$ is the minimizer of the population risk, *both within $\mathcal{A}_{\theta_0}^{\mathcal{C}_{\mathrm{T}}}$.*

Optimization error ($\varepsilon_{\mathrm{OPT}}$). The difference in performance between θ_{OPT} and θ_{ERM}, *i.e.* the error due to PGD, is controlled by the assumptions on the target loss landscape.

Estimation error ($\varepsilon_{\mathrm{EST}}$). By uniform convergence, θ_{ERM} performs similarly to $\bar{\theta}$, with the difference bounded using Rademacher complexity. By extension, θ_{OPT} performs similarly to $\bar{\theta}$.

Representation/approximation error ($\varepsilon_{\mathrm{REPR}}$). The difference $\mathcal{L}_\infty^{g^*}(g_{\bar{\theta}}) - \mathcal{L}_\infty^{g^*}(g^*)$ is bounded, on average, via the performance of the initialization θ_0 on the source tasks, using (ν,ε)-diversity. Since (6) is a finite-sample estimate of the desired bound, we again use uniform convergence arguments, which manifest as the second Rademacher complexity term.

5 Case Study: Two-Layer Neural Networks

We now illustrate our framework by instantiating our bounds in concrete settings. We focus on specializing Theorem 4.1 to the two-layer neural networks (this section) and logistic regression (provided in the appendix, Section E). For this section, we fix an activation function $\sigma : \mathbb{R} \to \mathbb{R}$.

Assumption 5.1. *For any $t \in [-2,2]$, $|\sigma'(t)| \le L$ and $|\sigma''(t)| \le \mu$. Furthermore, $\sigma(0) = 0$.*

Then, for a constant β and $\theta = (B, w)$, where $B \in \mathbb{R}^{d\times 2k}$ and $w \in \mathbb{R}^k$, we define the neural network $f_\theta^\beta(x) = \beta w^\top \sigma(B^\top x)$, where σ is applied elementwise.

5.1 Statistical Assumptions

Data Sampling. For all tasks, the inputs are sampled from a 1-norm-bounded distribution p. Furthermore, $q(\cdot \mid \mu)$ is generated as $\mu + \eta$ for some $O(1)$-bounded additive noise η, similar to Tripuraneni et al. (2020b).

Parameterizing the Tasks. We fix a representation $B^* \in \mathbb{R}^{d \times 2k}$ and a linear predictor $w_0^* \in \mathbb{R}^{2k}$. In this setting, we consider representation changes lying in a k-dimensional subspace, determined by an orthonormal[6] set of matrices $\Delta_1^*, \ldots, \Delta_k^*$ in $\mathbb{R}^{d \times 2k}$. Then, for each $t \in [T]$, we have two unit-norm vectors w_t^*, δ_t^* so that task t has parameter

$$\theta_t^* = \left(B^* + \frac{1}{\beta} \sum_{i \in [k]} (\delta_t^*)_i \Delta_i^* \ , \ w_0^* + \frac{1}{\beta} w_t^* \right).$$

We assume a diversity condition on (w_t^*, δ_t^*) similar in spirit to the linear version in Assumption 3.2, see Assumption D.2.

With appropriate assumptions on the initialization and large enough β, the source task predictors behave like their linearizations, *i.e.* there exists feature vectors ϕ_{B^*} and ψ_{B^*, w_0^*} so that

$$f_{\theta_t^*}(x) \approx \phi_{B^*}(x)^\top w_t^* + \left\langle \psi_{B^*, w_0^*}(x) \ , \ \sum_{i \in [k]} \delta_{t,i}^* \Delta_i^* \right\rangle.$$

We make use of these assumptions throughout for our result, see Assumption D.1 for the formal statement. Note that ϕ_{B^*} and ψ_{B^*, w_0^*} correspond to the "activation" and "gradient" features, respectively, that are empirically evaluated by Mu et al. (2020).

5.2 Training Procedure

We use the squared error loss to train in this setting. Additionally, we use the training procedure in Section 4.1 with the function class $\{f_\theta^\gamma\}$, with a minor caveat: we set $\gamma = \beta$ during source training, and $\gamma = O(\sqrt{n_{\mathrm{T}}})$ during target training.

With the procedure above, we now define the relevant feasible sets. We set Θ_0 be the set of initializations satisfying Assumption D.1. Finally, we set the constraint sets

$$\mathcal{C}_{\mathrm{S}} = \{(\Delta, w) \mid \|\Delta\|_F, \|w\|_2 \leq 1/\beta\} \quad \text{and} \quad \mathcal{C}_{\mathrm{T}}^\gamma = \left\{ (\Delta, w) \,\middle|\, \|\Delta\|_F^2 + \|w\|_2^2 \leq O(1/\gamma^2) \right\}.$$

5.3 Performance Guarantee

With the above assumptions in hand, we proceed with the performance guarantee.

Theorem 5.1 (Neural net performance bound)**.** *Assume that Assumptions 5.1, D.1 and D.2 hold. Then, if $n_{\mathrm{S}} \geq n_{\mathrm{T}}$, there exists a setting of the training parameters (see Section D) such that with probability at least $1 - \delta$, the iterates (θ_t) satisfy*

$$\mathbb{E}_{f^* \sim \rho} \left[\min_t \mathcal{L}_\infty^{f^*}(f_{\theta_t}^\gamma) - \mathcal{L}_\infty^{f^*}(f_{\theta^*}^\beta) \right] \lesssim L^2 \frac{k}{\sqrt{n_{\mathrm{T}}}} + L(L+\mu) \frac{k^{3/2}}{\sqrt{n_{\mathrm{S}} T}} + \left(\frac{\mu + L}{\beta} \right) L \sqrt{k}.$$

6 Conclusion

We have presented, to the best of our knowledge, the first statistical analysis of fine-tuning-based meta-learning. We demonstrate the success of such algorithms under the assumption of approximately shared representations between available tasks. In contrast, we show that "frozen representation" objectives analyzed by prior work fail under this weaker assumption.

An interesting line of future work is to determine ways to formulate useful shared structure among MDPs, *i.e.* formulate settings for which meta-reinforcement learning succeeds and results in improved regret bounds for downstream tasks.

[6] Orthonormal with respect to the Frobenius (*i.e.* entrywise) inner product on matrices.

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. *arXiv preprint arXiv:1710.03641*, 2017.

Alquier, P., Pontil, M., et al. Regret bounds for lifelong learning. In *Artificial Intelligence and Statistics*, pp. 261–269. PMLR, 2017.

Amit, R. and Meir, R. Meta-learning by adjusting priors based on extended pac-bayes theory. In *International Conference on Machine Learning*, pp. 205–214. PMLR, 2018.

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

Bai, Y., Chen, M., Zhou, P., Zhao, T., Lee, J. D., Kakade, S., Wang, H., and Xiong, C. How important is the train-validation split in meta-learning? *arXiv preprint arXiv:2010.05843*, 2020.

Balcan, M.-F., Blum, A., and Vempala, S. Efficient representations for lifelong learning and autoencoding. In *Conference on Learning Theory*, pp. 191–210. PMLR, 2015.

Baxter, J. A model of inductive bias learning. *Journal of artificial intelligence research*, 12:149–198, 2000.

Ben-David, S. and Borbely, R. S. A notion of task relatedness yielding provable multiple-task learning guarantees. *Machine learning*, 73(3):273–287, 2008.

Ben-David, S., Blitzer, J., Crammer, K., Kulesza, A., Pereira, F., and Vaughan, J. W. A theory of learning from different domains. *Machine learning*, 79(1):151–175, 2010.

Bengio, Y., Courville, A., and Vincent, P. Representation learning: A review and new perspectives. *IEEE transactions on pattern analysis and machine intelligence*, 35(8):1798–1828, 2013.

Bertinetto, L., Henriques, J. F., Torr, P. H. S., and Vedaldi, A. Meta-learning with differentiable closed-form solvers, 2019.

Bullins, B., Hazan, E., Kalai, A., and Livni, R. Generalize across tasks: Efficient algorithms for linear representation learning. In *Algorithmic Learning Theory*, pp. 235–246. PMLR, 2019.

Chen, T., Kornblith, S., Norouzi, M., and Hinton, G. A simple framework for contrastive learning of visual representations. In *International conference on machine learning*, pp. 1597–1607. PMLR, 2020.

Collins, L., Mokhtari, A., and Shakkottai, S. Why does maml outperform erm? an optimization perspective. *arXiv preprint arXiv:2010.14672*, 2020a.

Collins, L., Mokhtari, A., and Shakkottai, S. Task-robust model-agnostic meta-learning. *Advances in Neural Information Processing Systems*, 33, 2020b.

Denevi, G., Ciliberto, C., Stamos, D., and Pontil, M. Incremental learning-to-learn with statistical guarantees. *arXiv preprint arXiv:1803.08089*, 2018.

Denevi, G., Pontil, M., and Ciliberto, C. The advantage of conditional meta-learning for biased regularization and fine tuning. *Advances in Neural Information Processing Systems*, 33, 2020.

Denevi, G., Pontil, M., and Ciliberto, C. Conditional meta-learning of linear representations. *arXiv preprint arXiv:2103.16277*, 2021.

Du, S. S., Hu, W., Kakade, S. M., Lee, J. D., and Lei, Q. Few-shot learning via learning the representation, provably. *arXiv preprint arXiv:2002.09434*, 2020.

Duchi, J. Lecture notes for statistics 311/electrical engineering 377. *URL: https://stanford.edu/class/stats311/Lectures/full_notes. pdf. Last visited on*, 2:23, 2016.

Fallah, A., Mokhtari, A., and Ozdaglar, A. On the convergence theory of gradient-based model-agnostic meta-learning algorithms. In *International Conference on Artificial Intelligence and Statistics*, pp. 1082–1092. PMLR, 2020.

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

Finn, C., Rajeswaran, A., Kakade, S., and Levine, S. Online meta-learning. *arXiv preprint arXiv:1902.08438*, 2019.

Hanneke, S. and Kpotufe, S. A no-free-lunch theorem for multitask learning. *arXiv preprint arXiv:2006.15785*, 2020a.

Hanneke, S. and Kpotufe, S. On the value of target data in transfer learning. *arXiv preprint arXiv:2002.04747*, 2020b.

Jerfel, G., Grant, E., Griffiths, T. L., and Heller, K. Online gradient-based mixtures for transfer modulation in meta-learning. *arXiv preprint arXiv:1812.06080*, 2018.

Ji, K., Lee, J. D., Liang, Y., and Poor, H. V. Convergence of meta-learning with task-specific adaptation over partial parameters, 2020.

Khodak, M., Balcan, M.-F. F., and Talwalkar, A. S. Adaptive gradient-based meta-learning methods. In *Advances in Neural Information Processing Systems*, pp. 5917–5928, 2019.

Konobeev, M., Kuzborskij, I., and Szepesvári, C. On optimality of meta-learning in fixed-design regression with weighted biased regularization. *arXiv preprint arXiv:2011.00344*, 2020.

Kpotufe, S. and Martinet, G. Marginal singularity, and the benefits of labels in covariate-shift. In *Conference On Learning Theory*, pp. 1882–1886. PMLR, 2018.

Ledoux, M. and Talagrand, M. Probability in banach spaces: isoperimetry and processes. 1991.

Lee, J. D., Lei, Q., Saunshi, N., and Zhuo, J. Predicting what you already know helps: Provable self-supervised learning. *arXiv preprint arXiv:2008.01064*, 2020.

Lee, K., Maji, S., Ravichandran, A., and Soatto, S. Meta-learning with differentiable convex optimization, 2019.

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.

Lucas, J., Ren, M., Kameni, I., Pitassi, T., and Zemel, R. Theoretical bounds on estimation error for meta-learning. *arXiv preprint arXiv:2010.07140*, 2020.

Maurer, A. and Jaakkola, T. Algorithmic stability and meta-learning. *Journal of Machine Learning Research*, 6(6), 2005.

Maurer, A., Pontil, M., and Romera-Paredes, B. The benefit of multitask representation learning. *Journal of Machine Learning Research*, 17(81):1–32, 2016.

McMahan, B., Moore, E., Ramage, D., Hampson, S., and y Arcas, B. A. Communication-efficient learning of deep networks from decentralized data. In *Artificial Intelligence and Statistics*, pp. 1273–1282. PMLR, 2017.

Mendonca, R., Gupta, A., Kralev, R., Abbeel, P., Levine, S., and Finn, C. Guided meta-policy search. In *Advances in Neural Information Processing Systems*, pp. 9656–9667, 2019.

Mohri, M. and Medina, A. M. New analysis and algorithm for learning with drifting distributions. In *International Conference on Algorithmic Learning Theory*, pp. 124–138. Springer, 2012.

Mu, F., Liang, Y., and Li, Y. Gradients as features for deep representation learning. *arXiv preprint arXiv:2004.05529*, 2020.

Nichol, A., Achiam, J., and Schulman, J. On first-order meta-learning algorithms. *arXiv preprint arXiv:1803.02999*, 2018.

Pentina, A. and Lampert, C. A pac-bayesian bound for lifelong learning. In *International Conference on Machine Learning*, pp. 991–999. PMLR, 2014.

Raghu, M., Zhang, C., Kleinberg, J., and Bengio, S. Transfusion: Understanding transfer learning for medical imaging. *arXiv preprint arXiv:1902.07208*, 2019.

Rajeswaran, A., Finn, C., Kakade, S. M., and Levine, S. Meta-learning with implicit gradients. In *Advances in Neural Information Processing Systems*, pp. 113–124, 2019.

Salman, H., Ilyas, A., Engstrom, L., Kapoor, A., and Madry, A. Do adversarially robust imagenet models transfer better? *arXiv preprint arXiv:2007.08489*, 2020.

Saunshi, N., Zhang, Y., Khodak, M., and Arora, S. A sample complexity separation between non-convex and convex meta-learning. In *International Conference on Machine Learning*, pp. 8512–8521. PMLR, 2020.

Snell, J., Swersky, K., and Zemel, R. S. Prototypical networks for few-shot learning. *arXiv preprint arXiv:1703.05175*, 2017.

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

Tripuraneni, N., Jin, C., and Jordan, M. I. Provable meta-learning of linear representations. *arXiv preprint arXiv:2002.11684*, 2020a.

Tripuraneni, N., Jordan, M., and Jin, C. On the theory of transfer learning: The importance of task diversity. *Advances in Neural Information Processing Systems*, 33, 2020b.

Vershynin, R. Four lectures on probabilistic methods for data science. 2017.

Wang, H., Sun, R., and Li, B. Global convergence and generalization bound of gradient-based meta-learning with deep neural nets, 2020.

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

Checklist

1. For all authors...
   (a) Do the main claims made in the abstract and introduction accurately reflect the paper's contributions and scope? [Yes] Our introduction outline where each claim appears in the paper.
   (b) Did you describe the limitations of your work? [Yes] We have discussed extensively the assumptions that we require to establish our results, see Section 3.1, Section 4.2, Section 5.1, among others.
   (c) Did you discuss any potential negative societal impacts of your work? [No] As our work focuses on analyzing existing algorithms from a theoretical perspective, we do not foresee any negative societal impacts stemming from our work.
   (d) Have you read the ethics review guidelines and ensured that your paper conforms to them? [Yes]
2. If you are including theoretical results...
   (a) Did you state the full set of assumptions of all theoretical results? [Yes]
   (b) Did you include complete proofs of all theoretical results? [Yes]
3. If you ran experiments...
   (a) Did you include the code, data, and instructions needed to reproduce the main experimental results (either in the supplemental material or as a URL)? [Yes] A Jupyter notebook is provided to run the simulation outlined in Section B.3.
   (b) Did you specify all the training details (e.g., data splits, hyperparameters, how they were chosen)? [Yes] See Section B.3, as well as the provided Jupyter notebook.
   (c) Did you report error bars (e.g., with respect to the random seed after running experiments multiple times)? [No] Since we are simulating regression problems and can sample arbitrarily many times from the task distribution, we only report the mean in our experiments.
   (d) Did you include the total amount of compute and the type of resources used (e.g., type of GPUs, internal cluster, or cloud provider)? [No] Our experiments only involve simulations of simple settings that do not require extensive compute.
4. If you are using existing assets (e.g., code, data, models) or curating/releasing new assets...
   (a) If your work uses existing assets, did you cite the creators? [N/A]
   (b) Did you mention the license of the assets? [N/A]
   (c) Did you include any new assets either in the supplemental material or as a URL? [N/A]
   (d) Did you discuss whether and how consent was obtained from people whose data you're using/curating? [N/A]
   (e) Did you discuss whether the data you are using/curating contains personally identifiable information or offensive content? [N/A]
5. If you used crowdsourcing or conducted research with human subjects...
   (a) Did you include the full text of instructions given to participants and screenshots, if applicable? [N/A]
   (b) Did you describe any potential participant risks, with links to Institutional Review Board (IRB) approvals, if applicable? [N/A]
   (c) Did you include the estimated hourly wage paid to participants and the total amount spent on participant compensation? [N/A]