Learning-to-Learn Stochastic Gradient Descent with Biased Regularization

Giulia Denevi [1 2] **Carlo Ciliberto** [3 4] **Riccardo Grazzi** [1 4] **Massimiliano Pontil** [1 4]

Abstract

We study the problem of learning-to-learn: inferring a learning algorithm that works well on a family of tasks sampled from an unknown distribution. As class of algorithms we consider Stochastic Gradient Descent (SGD) on the true risk regularized by the square euclidean distance from a bias vector. We present an average excess risk bound for such a learning algorithm that quantifies the potential benefit of using a bias vector with respect to the unbiased case. We then propose a novel meta-algorithm to estimate the bias term online from a sequence of observed tasks. The small memory footprint and low time complexity of our approach makes it appealing in practice while our theoretical analysis provides guarantees on the generalization properties of the meta-algorithm on new tasks. A key feature of our results is that, when the number of tasks grows and their variance is relatively small, our learning-to-learn approach has a significant advantage over learning each task in isolation by standard SGD without a bias term. Numerical experiments demonstrate the effectiveness of our approach in practice.

1. Introduction

The problem of learning-to-learn (LTL) (Baxter, 2000; Thrun & Pratt, 1998) is receiving increasing attention in recent years, due to its practical importance (Finn et al., 2017; Franceschi et al., 2018; Ravi & Larochelle, 2017) and the theoretical challenge of statistically principled and efficient solutions (Alquier et al., 2017; Balcan et al., 2015; Maurer et al., 2016; Pentina & Lampert, 2014; Denevi et al., 2018a;b; Gupta & Roughgarden, 2017). The principal aim of LTL is to design a meta-learning algorithm to select a supervised learning algorithm that is well suited to learn tasks from a prescribed family. To highlight the difference between the meta-learning algorithm and the learning algorithm, throughout the paper we will refer to the latter as the *inner* or *within-task* algorithm.

The meta-algorithm is trained from a sequence of datasets, associated with different learning tasks sampled from a meta-distribution (also called *environment* in the literature). The performance of the selected inner algorithm is measured by the *transfer risk* (Baxter, 2000; Maurer, 2005), that is, the average risk of the algorithm, trained on a random dataset from the same environment. A key insight is that, when the learning tasks share specific similarities, the LTL framework provides a means to leverage such similarities and select an inner algorithm of low transfer risk.

In this work, we consider environments of linear regression or binary classification tasks and we assume that the associated weight vectors are all close to a common vector. Because of the increasing interest in low computational complexity procedures, we focus on the family of within-task algorithms given by Stochastic Gradient Descent (SGD) working on the regularized true risk. Specifically, motivated by the above assumption on the environment, we consider as regularizer the square distance of the weight vector to a bias vector, playing the role of a common mean among the tasks. Knowledge of this common mean can substantially facilitate the inner algorithm and the main goal of this paper is to design a meta-algorithm to learn a good bias that is supported by both computational and statistical guarantees.

Contributions. The first contribution of this work is to show that, when the variance of the weight tasks' vectors sampled from the environment is small, SGD regularized with the "right" bias yields a model with smaller error than its unbiased counterpart. The latter approach does not exploit the relatedness among the tasks, and it corresponds to learning the tasks in isolation – also known as independent task learning (ITL). The second and principal contribution of this work is to propose a meta-algorithm that estimates the bias term, so that the transfer risk of the corresponding SGD algorithm is as small as possible. We consider the setting in which we receive in input a sequence of datasets and we propose an online meta-algorithm which efficiently updates the bias term used by the inner SGD algorithm. Our meta-algorithm consists in applying a (meta) SGD algo-

[1]Istituto Italiano di Tecnologia, Genoa, Italy [2]University of Genoa, Genoa, Italy [3]Imperial College of London, London, United Kingdom [4]University College London, London, United Kingdom. Correspondence to: Giulia Denevi <giulia.denevi@iit.it>.

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

rithm to a proxy of the transfer risk, given by the expected minimum regularized empirical risk of a task. We provide a bound on the statistical performance of the biased inner SGD algorithm found by our procedure. It establishes that, when the number of observed tasks grows and the variance of the tasks' weight vectors is significantly smaller than their second moment, then, running the inner SGD algorithm with the estimated bias brings an improvement in comparison to learning the tasks in isolation with no bias. The bound is coherent with the state-of-the-art LTL analysis for other families of algorithms, but it applies for the first time to a fully online meta-algorithm. Our results holds for Lipschitz loss functions both in the regression and binary classification setting.

Our proof techniques combines ideas from online learning, stochastic and convex optimization, with tools from LTL. A key insight in our approach is to exploit the inner SGD algorithm to compute an approximate subgradient of the surrogate objective, in a such way that the degree of approximation can be controlled, without affecting the overall performance or the computational cost of the meta-algorithm.

Paper Organization. We start by recalling in Sec. 2 the basic concepts of LTL. In Sec. 3 we cast the problem of choosing a right bias term in SGD on the regularized objective in the LTL framework. Thanks to this formulation, in Sec. 4 we characterize the situations in which SGD with the right bias term is beneficial in comparison to SGD with no bias. In Sec. 5 we propose an online meta-algorithm to estimate the bias vector from a sequence of datasets and we analyze its statistical properties. In Sec. 6 we report on the empirical performance of the proposed approach while in Sec. 7 we discuss future research directions.

Previous Work. Online LTL (Alquier et al., 2017; Denevi et al., 2018a;b; Pentina & Urner, 2016) has received limited attention and is less developed than standard LTL approaches, in which the data are processed in one batch as opposed to incrementally, see for instance (Baxter, 2000; Maurer, 2009; Maurer et al., 2013; 2016; Pentina & Lampert, 2014). The idea of introducing a bias in the learning algorithm is not new, see e.g. (Denevi et al., 2018b; Kuzborskij & Orabona, 2017; Pentina & Lampert, 2014) and Sec. 3. In this work, we consider the family of inner SGD algorithms with biased regularization and we develop a theoretically grounded meta-learning algorithm to find the bias. Differently from others online methods (Alquier et al., 2017; Denevi et al., 2018a), our approach does not need to keep previous training points in memory and it runs online both across and within the tasks. As a result, both the low space and time complexity are the strengths of our method. We finally point out the recent related work by Khodak et al. (2019) and Finn et al. (2019), that was brought to our attention after completion of the present work.

2. Preliminaries

In this section, we recall the standard supervised (i.e. single-task) learning setting and the learning-to-learn setting.

We first introduce some notation used throughout this work. Let $\mathcal{Z} = \mathcal{X} \times \mathcal{Y}$ be the data space, where $\mathcal{X} \subset \mathbb{R}^d$ and $\mathcal{Y} \subseteq \mathbb{R}$ (regression) or $\mathcal{Y} = \{-1, +1\}$ (binary classification). We consider linear supervised learning tasks μ, namely distributions over $\mathcal{Z}$, parametrized by a weight vector $w \in \mathbb{R}^d$. We measure the performance by a loss function $\ell : \mathcal{Y} \times \mathcal{Y} \to \mathbb{R}_+$ such that, for any $y \in \mathcal{Y}$, $\ell(\cdot, y)$ is convex and closed. Finally, for any positive $k \in \mathbb{N}$, we let $[k] = \{1, \dots, k\}$ and, we denote by $\langle \cdot, \cdot \rangle$ and $\| \cdot \|$ the standard inner product and euclidean norm. In the rest of this work, when specified, we make the following assumptions.

Assumption 1 (Bounded Inputs)**.** *Let $\mathcal{X} \subseteq \mathcal{B}(0, R)$, where $\mathcal{B}(0, R) = \{x \in \mathbb{R}^d : \|x\| \leq R\}$, for some radius $R \geq 0$.*

Assumption 2 (Lipschitz Loss)**.** *Let $\ell(\cdot, y)$ be L-Lipschitz for any $y \in \mathcal{Y}$.*

For example, for any $y, \hat{y} \in \mathcal{Y}$, the absolute loss $\ell(\hat{y}, y) = |\hat{y} - y|$ and the hinge loss $\ell(\hat{y}, y) = \max\{0, 1 - y\hat{y}\}$ are both 1-Lipschitz. We now briefly recall the main notion of single-task learning.

2.1. Single-Task Learning

In standard linear supervised learning, the goal is to learn a linear functional relation $f_w : \mathcal{X} \to \mathcal{Y}$, $f_w(\cdot) = \langle \cdot, w \rangle$ between the input space $\mathcal{X}$ and the output space $\mathcal{Y}$. This target can be reformulated as that of finding a weight vector w_μ minimizing the *expected risk* (or true risk)

$$\mathcal{R}_\mu(w) = \mathbb{E}_{(x,y)\sim\mu}\, \ell\big(\langle x, w \rangle, y\big) \tag{1}$$

over the *entire* space $\mathbb{R}^d$. The expected risk measures the prediction error that the weight vector w incurs on average with respect to points sampled from the distribution μ. In practice, the task μ is unknown and only partially observed by a corresponding dataset of n i.i.d. points $Z_n = (z_i)_{i=1}^n \sim \mu^n$, where, for every $i \in [n]$, $z_i = (x_i, y_i) \in \mathcal{Z}$. In the sequel, we often use the more compact notation $Z_n = (X_n, \mathbf{y}_n)$, where $X_n \in \mathbb{R}^{n \times d}$ is the matrix containing the input vectors x_i as rows and $\mathbf{y}_n \in \mathbb{R}^n$ is the vector with entries given by the labels y_i. A *learning algorithm* is a function $A : \cup_{n\in\mathbb{N}} \mathcal{Z}^n \to \mathbb{R}^d$ that, given such a *training* dataset $Z_n \in \mathcal{Z}^n$, returns a "good" estimator, that is, in our case, a weight vector $A(Z_n) \in \mathbb{R}^d$, whose expected risk is small and tends to the minimum of Eq. (1) as n increases.

2.2. Learning-to-Learn (LTL)

In the LTL framework, we assume that each learning task μ we observe is sampled from an *environment* ρ, that is a (meta-)distribution on the set of probability distributions on

$\mathcal{Z}$. The goal is to select a learning algorithm (hence the name *learning-to-learn*) that is well suited to the environment.

Specifically, we consider the following setting. We receive a stream of tasks $\mu_1, \dots, \mu_T$, which are independently sampled from the ρ and only partially observed by corresponding i.i.d. datasets $Z_n^{(1)}, \dots, Z_n^{(T)}, \dots$ each formed by n datapoints. Starting from these datasets, we wish to learn an algorithm A, such that, when we apply it on a new dataset (composed by n points) sampled from a new task $\mu \sim \rho$, the corresponding true risk is low. We reformulate this target into requiring that algorithm A trained with n points[1] over the environment ρ, has small *transfer risk*

$$\mathcal{E}_n(A) = \mathbb{E}_{\mu\sim\rho}\, \mathbb{E}_{Z_n\sim\mu^n}\, \mathcal{R}_\mu(A(Z_n)). \tag{2}$$

The transfer risk measures the expected true risk that the inner algorithm A, trained on the dataset Z_n, incurs *on average with respect to the distribution of tasks μ sampled from ρ*. Therefore, the process of learning a learning algorithm is a meta-learning one, in that the inner learning algorithm is applied to tasks from the environment and then chosen from a sequence of training tasks (datasets) in attempt to minimize the transfer risk.

3. SGD on the Biased Regularized Risk

In this section, we introduce the LTL framework for the family of within-task algorithms we analyze in this work.

We consider a family of learning algorithms A_h parametrized by a bias vector $h \in \mathbb{R}^d$. The idea of introducing a bias in a specific family of learning algorithms is not new in the LTL literature, see e.g. (Denevi et al., 2018b; Kuzborskij & Orabona, 2017; Pentina & Lampert, 2014) and references therein. A natural choice is given by regularized empirical risk minimization, in which we introduce a bias vector in the square norm regularizer – which we simply refer to as ERM throughout – namely

$$A_h^{\mathrm{ERM}}(Z_n) \equiv w_h(Z_n) = \operatorname*{argmin}_{w\in\mathbb{R}^d} \mathcal{R}_{Z_n,h}(w), \tag{3}$$

where, for any $w, h \in \mathbb{R}^d$, $\lambda > 0$, we have defined the empirical error and its biased regularized version as

$$\begin{aligned} \mathcal{R}_{Z_n}(w) &= \frac{1}{n}\sum_{k=1}^n \ell_k\big(\langle x_k, w\rangle\big) \\ \mathcal{R}_{Z_n,h}(w) &= \mathcal{R}_{Z_n}(w) + \frac{\lambda}{2}\|w-h\|^2. \end{aligned} \tag{4}$$

Intuitively, if the weight vectors w_μ of the tasks sampled from ρ are close to each other, then running ERM with $h = \mathsf{m} \equiv \mathbb{E}_{\mu\sim\rho} w_\mu$ should have a smaller transfer risk than running ERM with, for instance, $h = 0$. We make this statement precise in Sec. 4. Recently, a number of papers have considered how to learn a good bias h in a LTL setting, see e.g. (Pentina & Lampert, 2014; Denevi et al., 2018b). However, one drawback of these works is that they assume the ERM solution to be known exactly, without leveraging the interplay between the optimization and the generalization error. Furthermore, in LTL settings, data naturally arise in an online manner, both *between* and *within* tasks. Hence, an ideal LTL approach should focus on inner algorithms processing one single data point at the time.

Algorithm 1 Within-Task Algorithm: SGD on the Biased Regularized True Risk

Input $\lambda > 0$ regularization parameter, h bias, μ task

Initialization ${w_h}^{(1)} = h$

For $k = 1$ to n

Receive $(x_k, y_k) \sim \mu$

Build $\ell_{k,h}(\cdot) = \ell_k(\langle x_k, \cdot\rangle) + \frac{\lambda}{2}\|\cdot - h\|^2$

Define $\gamma_k = 1/(k\lambda)$

Compute $u_k' \in \partial\ell_k(\langle x_k, {w_h}^{(k)}\rangle)$

Define $s_k = x_k u_k' + \lambda({w_h}^{(k)} - h) \in \partial\ell_{k,h}({w_h}^{(k)})$

Update ${w_h}^{(k+1)} = {w_h}^{(k)} - \gamma_k s_k$

Return $({w_h}^{(k)})_{k=1}^{n+1}$, $\bar{w}_h = \frac{1}{n}\sum_{i=1}^n {w_h}^{(i)}$

Motivated by the above reasoning, in this work, we propose to analyze an online learning algorithm that is computationally and memory efficient while retaining (on average with respect to the sampling of the data) the *same statistical guarantees* of the more expensive ERM estimator. Specifically, for a training dataset $Z_n \sim \mu^n$, a regularization parameter $\lambda > 0$ and a bias vector $h \in \mathbb{R}^d$, we consider the learning algorithm defined as

$$A_h^{\mathrm{SGD}}(Z_n) \equiv \bar{w}_h(Z_n), \tag{5}$$

where, $\bar{w}_h(Z_n)$ is the average of the first n iterations of Alg. 1, in which, for any $k \in [n]$, we have introduced the notation $\ell_k(\cdot) = \ell(\cdot, y_k)$.

Alg. 1 coincides with online subgradient algorithm applied to the strongly convex function $\mathcal{R}_{Z_n,h}$. Moreover, thanks to the assumption that $Z_n \sim \mu^n$, Alg. 1 is equivalent to SGD applied to the regularized true risk

$$\mathcal{R}_{\mu,h}(w) = \mathcal{R}_\mu(w) + \frac{\lambda}{2}\|w-h\|^2. \tag{6}$$

[1] In order to simplify the presentation, we assume that all datasets are composed by the same number of points n. The general setting can be addressed by introducing the slightly different notion of transfer risk $\mathcal{E}(A) = \mathbb{E}_{(n,\mu)\sim\rho}\, \mathbb{E}_{Z_n\sim\mu^n}\, \mathcal{R}_\mu(A(Z_n))$.

Relying on a standard online-to-batch analysis, see e.g. (Cesa-Bianchi et al., 2004; Hazan, 2016) and references therein, it is easy to link the true error of such an algorithm with the minimum of the regularized empirical risk, that is, $\mathcal{R}_{Z_n,h}(w_h(Z_n))$. This fact is reported in the proposition below and it will be often used in our subsequent statistical analysis. We give a proof in App. F for completeness.

Proposition 1. *Let Asm. 1 and Asm. 2 hold and let $\bar{w}_h$ be the output of Alg. 1. Then, we have that*

$$\mathbb{E}_{Z_n\sim\mu^n}\left[\mathcal{R}_\mu\big(\bar{w}_h(Z_n)\big) - \mathcal{R}_{Z_n,h}(w_h(Z_n))\right] \le c_{n,\lambda} \qquad c_{n,\lambda} = \frac{2R^2L^2\big(\log(n)+1\big)}{\lambda n}. \tag{7}$$

We remark that at this level of the analysis, one may avoid the logarithmic factor in the above bound, see e.g. (Shamir & Zhang, 2013; Rakhlin et al., 2012; Lacoste-Julien et al., 2012). However, in order to not complicate our presentation and proofs, we avoid this refinement of the analysis.

In the next section we study the impact on the bias vector on the statistical performance of the inner algorithm. Specifically, we investigate circumstances under which there is an advantage in perturbing the regularization in the objective used by the algorithm with an appropriate ideal bias term h, as opposed to fix $h = 0$. Throughout the paper, we refer to the choice $h = 0$ as independent task learning (ITL), although strictly speaking, when h is fixed in advanced, then, SGD is applied on each task independently regardless of the value of h. Then, in Sec. 5 we address the question of estimating this appropriate bias from the data.

4. The Advantage of the Right Bias Term

In this section, we study the statistical performance of the model $\bar{w}_h$ returned by Alg. 1, on average with respect to the tasks sampled from the environment ρ, for different choices of the bias vector h. To present our observations, we require, for any $\mu \sim \rho$, that the corresponding true risk admits minimizers and we denote by w_μ the minimum norm minimizer[2]. With these ingredients, we introduce the oracle

$$\mathcal{E}_\rho = \mathbb{E}_{\mu\sim\rho}\, \mathcal{R}_\mu(w_\mu),$$

representing the averaged minimum error over the environment of tasks, and, for a candidate bias h, we give a bound on the quantity $\mathcal{E}(\bar{w}_h) - \mathcal{E}_\rho$. This gap coincides with the averaged excess risk of algorithm Alg. 1 with bias h over the environment of tasks, that is

$$\mathcal{E}_n(\bar{w}_h) - \mathcal{E}_\rho = \mathbb{E}_{\mu\sim\rho}\, \mathbb{E}_{Z_n\sim\mu^n}\left[\mathcal{R}_\mu\big(\bar{w}_h(Z_n)\big) - \mathcal{R}_\mu(w_\mu)\right].$$

[2]This choice is made in order to simplify our presentation. However, our analysis holds for different choices of a minimizer w_μ, which may potentially improve our bounds.

Hence, this quantity is an indicator of the performance of the bias h with respect to our environment. In the rest of this section, we study the above gap for a bias h which is fixed and does not depend on the data. For this purpose, we introduce the notation

$$\mathrm{Var}_h^2 = \frac{1}{2}\,\mathbb{E}_{\mu\sim\rho}\,\|w_\mu - h\|^2 \tag{8}$$

and we observe that

$$\mathsf{m} \equiv \mathbb{E}_{\mu\sim\rho} w_\mu = \operatorname*{argmin}_{h\in\mathbb{R}^d}\ \mathrm{Var}_h^2. \tag{9}$$

Theorem 2 (Excess Transfer Risk Bound for a Fixed Bias h)**.** *Let Asm. 1 and Asm. 2 hold and let $\bar{w}_h$ be the output of Alg. 1 with regularization parameter*

$$\lambda = \frac{RL}{\mathrm{Var}_h}\sqrt{\frac{2\big(\log(n)+1\big)}{n}}. \tag{10}$$

Then, the following bound holds

$$\mathcal{E}_n(\bar{w}_h) - \mathcal{E}_\rho \le \mathrm{Var}_h\, 2RL\sqrt{\frac{2\big(\log(n)+1\big)}{n}}. \tag{11}$$

Proof. For $\mu\sim\rho$, consider the following decomposition

$$\mathbb{E}_{Z_n\sim\mu^n}\left[\mathcal{R}_\mu(\bar{w}_h(Z_n)) - \mathcal{R}_\mu(w_\mu)\right] \le \mathrm{A} + \mathrm{B}, \tag{12}$$

where A and B are respectively defined by

$$\begin{aligned}\mathrm{A} &= \mathbb{E}_{Z_n\sim\mu^n}\left[\mathcal{R}_\mu(\bar{w}_h(Z_n)) - \mathcal{R}_{Z_n,h}(w_h(Z_n))\right]\\ \mathrm{B} &= \mathbb{E}_{Z_n\sim\mu^n}\left[\mathcal{R}_{Z_n,h}(w_h(Z_n)) - \mathcal{R}_\mu(w_\mu)\right].\end{aligned} \tag{13}$$

In order to bound the term A, we use Prop. 1. Regarding the term B, we exploit the definition of the ERM algorithm and the fact that, since w_μ does not depend on Z_n, then $\mathcal{R}_{\mu,h}(w_\mu) = \mathbb{E}_{Z_n\sim\mu^n}\mathcal{R}_{Z_n,h}(w_\mu)$. Consequently, we can upper bound the term B as

$$\begin{aligned}&\mathbb{E}_{Z_n\sim\mu^n}\left[\mathcal{R}_{Z_n,h}(w_h(Z_n)) - \mathcal{R}_{\mu,h}(w_\mu)\right] + \frac{\lambda}{2}\|w_\mu - h\|^2\\ &= \mathbb{E}_{Z_n\sim\mu^n}\left[\mathcal{R}_{Z_n,h}(w_h(Z_n)) - \mathcal{R}_{Z_n,h}(w_\mu)\right] + \frac{\lambda}{2}\|w_\mu - h\|^2\\ &\le \frac{\lambda}{2}\|w_\mu - h\|^2.\end{aligned} \tag{14}$$

The desired statement follows by combining the above bounds on the two terms, taking the average with respect to $\mu\sim\rho$ and optimizing over λ. ∎

Thm. 2 shows that the strength of the regularization that one should use in the within-task algorithm Alg. 1 is inversely

proportional to both the variance of the bias h and the number of points in the datasets. This is exactly in line with the LTL aim: when solving each task is difficult, knowing a priori a good bias can bring a substantial benefit over learning with no bias. To further investigate this point, in the following corollary, we specialize Thm. 2 to two particular choices of the bias h. The first choice we make is $h = 0$, which coincides, as remarked earlier, with learning each task independently, while the second choice considers an ideal bias, namely, assuming that the transfer risk admits minimizer, we set $h = h_n \in \operatorname{argmin}_{h\in\mathbb{R}^d} \mathcal{E}_n(\bar{w}_h)$.

Corollary 3 (Excess Transfer Risk Bound for ITL and the Oracle)**.** *Let Asm. 1 and Asm. 2 hold.*

1. **Independent Task Learning.** *Let $\bar{w}_0$ be the output of Alg. 1 with bias $h = 0$ and regularization parameter as in Eq. (10) with $h = 0$. Then,*

$$\mathcal{E}_n(\bar{w}_0) - \mathcal{E}_\rho \leq \mathrm{Var}_0\, 2RL \sqrt{\frac{2\big(\log(n)+1\big)}{n}}.$$

2. **The Oracle.** *Let $\bar{w}_{h_n}$ be the output of Alg. 1 with bias $h = h_n$ and regularization parameter as in Eq. (10) with $h = \mathsf{m}$. Then,*

$$\mathcal{E}_n(\bar{w}_{h_n}) - \mathcal{E}_\rho \leq \mathrm{Var}_{\mathsf{m}}\, 2RL \sqrt{\frac{2\big(\log(n)+1\big)}{n}}.$$

Proof. The proof of the first statement follows directly from the application of Thm. 2 with $h = 0$. The second statement is a direct consequence of the definition of h_n implying $\mathcal{E}_n(\bar{w}_{h_n}) - \mathcal{E}_\rho \leq \mathcal{E}_n(\bar{w}_{\mathsf{m}}) - \mathcal{E}_\rho$ and the application of Thm. 2 with $h = \mathsf{m}$ on the second term. ■

From the previous bounds we can observe that, using the bias $h = h_n$ in the regularizer brings a substantial benefit with respect to the unbiased case when the number of points n in each dataset in not very large (hence learning each task is quite difficult) and the variance of the weight tasks' vectors sampled from the environment is much smaller than their second moment, i.e. when

$$\mathrm{Var}_{\mathsf{m}}^2 = \frac{1}{2}\,\mathbb{E}_{\mu\sim\rho}\,\|w_\mu - \mathsf{m}\|^2 \ll \frac{1}{2}\,\mathbb{E}_{\mu\sim\rho}\,\|w_\mu\|^2 = \mathrm{Var}_0^2.$$

Driven by this observation, when the environment of tasks satisfies the above characteristics, we would like to take advantage of this tasks' similarity. But, since in practice we are not able to explicitly compute h_n, in the following we propose an efficient online LTL approach to estimate the bias directly from the observed sequence of datasets.

5. Estimating the Bias

In this section, we study the problem of designing an estimator for the bias vector that is computed *incrementally* from a set of observed T tasks.

5.1. The Meta-Objective

Since direct optimization of the transfer risk is not feasible, a standard strategy used in LTL consists in introducing a proxy objective that is easier to handle, see e.g. (Maurer, 2005; 2009; Maurer et al., 2013; 2016; Denevi et al., 2018a;b). In this paper, motivated by Prop. 1, according to which

$$\mathbb{E}_{Z_n\sim\mu^n}\left[\mathcal{R}_\mu\big(\bar{w}_h(Z_n)\big)\right] \leq \mathbb{E}_{Z_n\sim\mu^n}\left[\mathcal{R}_{Z_n,h}(w_h(Z_n))\right] + \frac{2R^2L^2\big(\log(n)+1\big)}{\lambda n},$$

we substitute in the definition of the transfer risk the true risk of the algorithm $\mathcal{R}_\mu\big(\bar{w}_h(Z_n)\big)$ with the minimum of the regularized empirical risk

$$\mathcal{L}_{Z_n}(h) = \min_{w\in\mathbb{R}^d} \mathcal{R}_{Z_n,h}(w) = \mathcal{R}_{Z_n,h}(w_h(Z_n)). \tag{15}$$

This leads us to the following proxy for the transfer risk

$$\hat{\mathcal{E}}_n(h) = \mathbb{E}_{\mu\sim\rho}\,\mathbb{E}_{Z_n\sim\mu^n}\,\mathcal{L}_{Z_n}(h). \tag{16}$$

Some remarks about this choice are in order. First, convexity is usually a rare property in LTL. In our case, as described in the following proposition, the definition of the function $\mathcal{L}_{Z_n}$ as the partial minimum of a jointly convex function, ensures convexity and other nice properties, such as differentiability and a closed expression of its gradient.

Proposition 4 (Properties of $\mathcal{L}_{Z_n}$)**.** *The function $\mathcal{L}_{Z_n}$ in Eq. (15) is convex and λ-smooth over $\mathbb{R}^d$. Moreover, for any $h \in \mathbb{R}^d$, its gradient is given by the formula*

$$\nabla\mathcal{L}_{Z_n}(h) = -\lambda\big(w_h(Z_n) - h\big), \tag{17}$$

where $w_h(Z_n)$ is the ERM algorithm in Eq. (3). Finally, when Asm. 1 and Asm. 2 hold, $\mathcal{L}_{Z_n}$ is LR-Lipschitz.

The above statement is a known result in the optimization community, see e.g. (Bauschke & Combettes, 2011, Prop. 12.29) and App. C for more details. In order to minimize the proxy objective in Eq. (16), one standard choice done in stochastic optimization, and also adopted in this work, is to use first-order methods, requiring the computation of an unbiased estimate of the gradient of the stochastic objective. In our case, according to the above proposition, this step would require computing the minimizer of the regularized empirical problem in Eq. (15) exactly. A key observation of our work is to show below that we can easily design a "satisfactory" approximation (see the last paragraph

in Sec. 5) of its gradient, just substituting the minimizer $w_h(Z_n)$ in the expression of the gradient in Eq. (17) with the last iterate $w_h{}^{(n+1)}(Z_n)$ of Alg. 1. An important aspect to stress here is the fact that this strategy does not require any additional computational effort. Formally, this reasoning is linked to the concept of ϵ-subgradient of a function. We recall that, for a given convex, proper and closed function f and for a given point $\hat{h} \in \mathrm{Dom}(f)$ in its domain, u is an ϵ-subgradient of f at $\hat{h}$, if, for any h, $f(h) \geq f(\hat{h}) + \langle u, h - \hat{h} \rangle - \epsilon$.

Proposition 5 (An ϵ-Subgradient for $\mathcal{L}_{Z_n}$)**.** *Let $w_h{}^{(n+1)}(Z_n)$ be the last iterate of Alg. 1. Then, under Asm. 1 and Asm. 2, the vector*

$$\hat{\nabla}\mathcal{L}_{Z_n}(h) = -\lambda\big(w_h{}^{(n+1)}(Z_n) - h\big) \tag{18}$$

is an ϵ-subgradient of $\mathcal{L}_{Z_n}$ at point h, where ϵ is such that

$$\mathbb{E}_{Z_n \sim \mu^n}\,[\epsilon] \leq \frac{2R^2L^2\big(\log(n)+1\big)}{\lambda n}. \tag{19}$$

Moreover, introducing $\Delta_{Z_n}(h) = \nabla\mathcal{L}_{Z_n}(h) - \hat{\nabla}\mathcal{L}_{Z_n}(h)$,

$$\mathbb{E}_{Z_n \sim \mu^n}\,\big\|\Delta_{Z_n}(h)\big\|^2 \leq \frac{4R^2L^2\big(\log(n)+1\big)}{n}. \tag{20}$$

The above result is a key tool in our analysis. The proof requires some preliminaries on the ϵ-subdifferential of a function (see App. A) and introducing the dual formulation of both the within-task learning problem and Alg. 1 (see App. B and App. E, respectively). With these two ingredients, the proof of the statement is deduced in App. E.3 by the application of a more general result reported in App. D, describing how an ϵ-minimizer of the dual of the within-task learning problem can be exploited in order to build an ϵ-subgradient of the meta-objective function $\mathcal{L}_{Z_n}$. We stress that this result could be applied to more general class of algorithms, going beyond Alg. 1 considered here.

5.2. The Meta-Algorithm to Estimate the Bias h

In order to estimate the bias h from the data, we apply SGD to the stochastic function $\hat{\mathcal{E}}_n$ introduced in Eq. (16). More precisely, in our setting, the sampling of a "meta-point" corresponds to the incremental sampling of a dataset from the environment[3]. We refer to Alg. 2 for more details. In particular, we propose to take the estimator $\bar{h}_T$ obtained by averaging the iterations returned by Alg. 2. An important feature to stress here is the fact that the meta-algorithm uses ϵ-subgradients of the function $\mathcal{L}_{Z_n}$ which are computed as described above. Specifically, for any $t \in [T]$, we define

$$\hat{\nabla}\mathcal{L}_{Z_n^{(t)}}(h^{(t)}) = -\lambda\big(w_{h^{(t)}}^{(n+1)}(Z_n^{(t)}) - h^{(t)}\big), \tag{21}$$

[3]More precisely we first sample a distribution μ from ρ and then a dataset $Z_n \sim \mu^n$.

Algorithm 2 Meta-Algorithm, SGD on $\hat{\mathcal{E}}$ with ϵ-Subgradients

Input $\gamma > 0$ step size, $\lambda > 0$ inner regularization parameter, ρ meta-distribution

Initialization $h^{(1)} = 0 \in \mathbb{R}^d$

For $t = 1$ to T

Receive $\mu_t \sim \rho$, $Z_n^{(t)} \sim \mu_t^n$

Run the inner algorithm Alg. 1 and approximate the gradient $\hat{\nabla}^{(t)} \approx \nabla^{(t)}$ by Eq. (21)

Update $h^{(t+1)} = h^{(t)} - \gamma\hat{\nabla}^{(t)}$

Return $(h^{(t)})_{t=1}^{T+1}$ and $\bar{h}_T = \dfrac{1}{T}\displaystyle\sum_{t=1}^{T} h^{(t)}$

where $w_{h^{(t)}}^{(n+1)}$ is the last iterate of Alg. 1 applied with the current bias $h^{(t)}$ and the dataset $Z_n^{(t)}$. To simplify the presentation, throughout this work, we use the short-hand notation

$$\mathcal{L}_t(\cdot) = \mathcal{L}_{Z_n^{(t)}}(\cdot),\ \nabla^{(t)} = \nabla\mathcal{L}_t(h^{(t)}),\ \hat{\nabla}^{(t)} = \hat{\nabla}\mathcal{L}_t(h^{(t)}).$$

Some technical observations follows. First, we stress that Alg. 2 processes one single instance at the time, without the need to store previously encountered data points, neither across the tasks nor within them. Second, the implementation of Alg. 2 does not require computing the meta-objective $\mathcal{L}_{Z_n}$, which would increase the computational effort of the entire scheme. The rest of this section is devoted to the statistical analysis of Alg. 2.

5.3. Statistical Analysis of the Meta-Algorithm

In the following theorem we study the statistical performance of the bias $\bar{h}_T$ returned by Alg. 2. More precisely we bound the excess transfer risk of the inner SGD algorithm run with this biased term learned by the meta-algorithm.

Theorem 6 (Excess Transfer Risk Bound for the Bias $\bar{h}_T$ Estimated by Alg. 2)**.** *Let Asm. 1 and Asm. 2 hold and let $\bar{h}_T$ be the output of Alg. 2 with step size*

$$\gamma = \frac{\sqrt{2}\|\mathsf{m}\|}{LR}\sqrt{\Big(T\Big(1 + \frac{4\big(\log(n)+1\big)}{n}\Big)\Big)^{-1}}. \tag{22}$$

Let $\bar{w}_{\bar{h}_T}$ be the output of Alg. 1 with bias $h = \bar{h}_T$ and regularization parameter

$$\lambda = \frac{2RL}{\mathrm{Var}_{\mathsf{m}}}\sqrt{\frac{\log(n)+1}{n}}. \tag{23}$$

Then, the following bound holds

$$\mathbb{E}\left[\mathcal{E}_n(\bar{w}_{\bar{h}_T})\right] - \mathcal{E}_\rho \leq \mathrm{Var}_{\mathsf{m}}\, 4RL\sqrt{\frac{\log(n)+1}{n}} + \|\mathsf{m}\|RL\sqrt{2\Big(1+\frac{4\big(\log(n)+1\big)}{n}\Big)\frac{1}{T}}$$

where the expectation above is with respect to the sampling of the datasets $Z_n^{(1)}, \ldots, Z_n^{(T)}$ from the environment ρ.

Proof. We consider the following decomposition

$$\mathbb{E}\left[\mathcal{E}_n(\bar{w}_{\bar{h}_T})\right] - \mathcal{E}_\rho \leq \mathrm{A} + \mathrm{B} + \mathrm{C}, \tag{24}$$

where

$$\begin{aligned} \mathrm{A} &= \mathcal{E}_n(\bar{w}_{\bar{h}_T}) - \hat{\mathcal{E}}_n(\bar{h}_T) \\ \mathrm{B} &= \mathbb{E}\,\hat{\mathcal{E}}_n(\bar{h}_T) - \hat{\mathcal{E}}_n(\mathsf{m}) \\ \mathrm{C} &= \hat{\mathcal{E}}_n(\mathsf{m}) - \mathcal{E}_\rho. \end{aligned} \tag{25}$$

Now, in order to bound the term A, noting that

$$\mathrm{A} = \mathbb{E}_{\mu\sim\rho}\,\mathbb{E}_{Z_n\sim\mu^n}\left[\mathcal{R}_\mu\big(\bar{w}_{\bar{h}_T}(Z_n)\big) - \mathcal{R}_{Z_n,\bar{h}_T}(w_{\bar{h}_T}(Z_n))\right],$$

we use Prop. 1 with $h = \bar{h}_T$ and, then, we take the average on $\mu \sim \rho$. As regards the term C, we apply the inequality given in Eq. (14) with $h = \mathsf{m}$ and we again average with respect to $\mu \sim \rho$. Finally, the term B is the convergence rate of Alg. 2 and its study requires analyzing the error that we introduce in the meta-gradients by Prop. 5. The bound we use for this term described in Prop. 22 (see App. G) with $\hat{h} = \mathsf{m}$. The result now follows by combining the bounds on the three terms and optimizing over λ. ∎

The bound stated in Thm. 6 with respect to the mean m holds also for a generic bias vector $h \in \mathbb{R}^d$. In particular, the choice of $h = 0$ and the corresponding step-size γ describe the setting in which the meta-algorithm returns the ITL estimator $h = 0$. In such a case, we recover the rate in Cor. 3 for ITL (up to a contant 2).

In addition, the above bound is coherent with the state-of-the-art LTL bounds given in other papers studying other variants of Ivanov or Tikhonov regularized empirical risk minimization algorithms, see e.g. (Maurer, 2005; 2009; Maurer et al., 2013; 2016). Specifically, in our case, the bound has the form

$$\mathcal{O}\Big(\frac{\mathrm{Var}_{\mathsf{m}}}{\sqrt{n}}\Big) + \mathcal{O}\Big(\frac{1}{\sqrt{T}}\Big), \tag{26}$$

where $\mathrm{Var}_{\mathsf{m}}$ reflects the advantage in exploiting the relatedness among the tasks sampled from the environment ρ. More precisely, in Sec. 4 we noticed that, if the variance of the weight vectors of the tasks sampled from our environment is significantly smaller than their second moment, running Alg. 1 with the ideal bias $h = h_n$ on a future task brings a significant improvement in comparison to the unbiased case. One natural question arising at this point of the presentation is whether, under the same conditions on the environment, the same improvement is obtained by running Alg. 1 with the bias vector $h = \bar{h}_T$ returned by our online meta-algorithm in Alg. 2. Looking at the bound in Thm. 6, we can say that, when the number of training tasks T used to estimate the bias $\bar{h}_T$ is sufficiently large, the above question has a positive answer and our LTL approach is effective.

In order to have also a more precise benchmark for the biased setting considered in this work, in App. H we have repeated the statistical study described in the paper also for the more expensive ERM algorithm described in Eq. (3). In this case, we assume to have an oracle providing us with this exact estimator, ignoring any computational costs. As before, we have performed the analysis both for a fixed bias and the one estimated from the data via Alg. 2 (in this case, Alg. 2 is assumed to run with exact meta-gradients). Looking at the results reported in App. H, we immediately see that, up to constants and logarithmic factors, the LTL bounds we have stated in the paper for the low-complexity SGD family are equivalent to those in App. H for the more expensive ERM family.

All the above facts justify the informal statement given before Prop. 5 according to which the trick used to compute the approximation of the meta-gradient by using the last iterate of the inner algorithm, not only, does not require additional effort, but it is also accurate enough from the statistical view point, matching a state-of-the-art bound for more expensive within-task algorithms based on ERM.

6. Experiments

In this section, we test the effectiveness of the LTL approach proposed in this paper on synthetic and real data[4]. In all experiments, the regularization parameter λ and the step-size γ were tuned by validation, see App. I for more details.

Synthetic Data. We considered two different settings, regression with the absolute loss and binary classification with the hinge loss. In both cases, we generated an environment of tasks in which SGD with the right bias is expected to bring a substantial benefit in comparison to the unbiased case. Motivated by our observations in Sec. 4, we generated linear tasks with weight vectors characterized by a variance which is significantly smaller than their second moment. Specifically, for each task μ, we created a weight vector w_μ from a Gaussian distribution with mean m given by the

[4]Code available at https://github.com/prolearner/onlineLTL

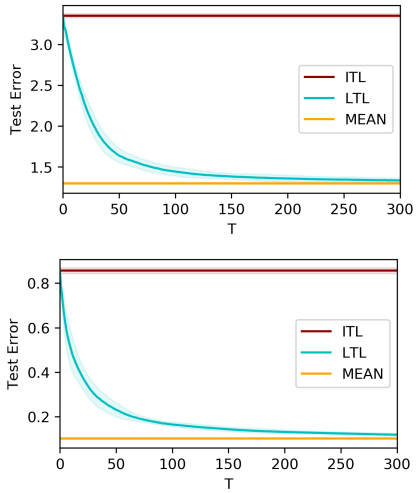

Figure 1. **Synthetic Data.** Test performance of different bias with respect to an increasing number of tasks. (Top) Regression with absolute loss. (Bottom) Classification with hinge loss. The results are averaged over 10 independent runs (datasets generations).

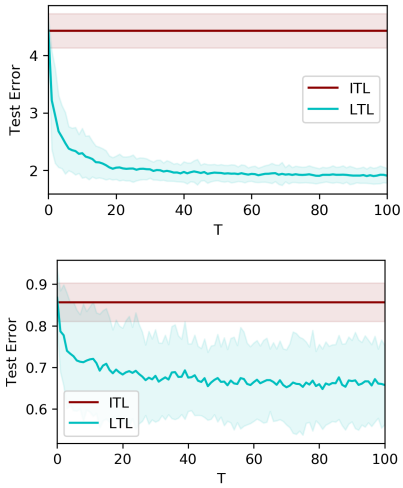

Figure 2. **Real Data.** Test performance of different bias with respect to an increasing number of tasks. (Top) Lenk Dataset Regression. (Bottom) Lenk Dataset Classification. The results are averaged over 30 independent runs (datasets generations).

vector in $\mathbb{R}^d$ with all components equal to 4 and standard deviation $\mathrm{Var}_{\mathsf{m}} = 1$. Each task corresponds to a dataset $(x_i, y_i)_{i=1}^n$, $x_i \in \mathbb{R}^d$ with $n = 10$ and $d = 30$. In the regression case, the inputs were uniformly sampled on the unit sphere and the labels were generated as $y = \langle x, w_\mu \rangle + \epsilon$, with ϵ sampled from a zero-mean Gaussian distribution, with standard deviation chosen to have signal-to-noise ratio equal to 10 for each task. In the classification case, the inputs were uniformly sampled on the unit sphere, excluding those points with margin $|\langle x, w_\mu \rangle|$ smaller than 0.5 and the binary labels were generated as a logistic model, $\mathbb{P}(y = 1) = \left(1 + 10 \exp(-\langle x, w_\mu \rangle)\right)^{-1}$. In Fig. 1 we report the performance of Alg. 1 with different choices of the bias: $h = \bar{h}_T$ (our LTL estimator resulting from Alg. 2), $h = 0$ (ITL) and $h = \mathsf{m}$, a reasonable approximation of the oracle minimizing the transfer risk. The plots confirm our theoretical findings: estimating the bias with our LTL approach leads to a substantial benefits with respect to the unbiased case, as the number of the observed training tasks increases.

Real Data. We run experiments on the computer survey data from (Lenk et al., 1996), in which 180 people (tasks) rated the likelihood of purchasing one of 20 different personal computers ($n = 8$). The input represents 13 different computer characteristics (price, CPU, RAM, etc.) while the output is an integer rating from 0 to 10. Similarly to the synthetic data experiments, we consider a regression setting with the absolute loss and a classification setting. In the latter case each task is to predict whether the rating is above 5. We compare the LTL bias with ITL. The results are reported in Fig. 2. The figures above are in line with the results obtained on synthetic experiments, indicating that the bias LTL framework proposed in this work is effective for this dataset. Moreover, the results for regression are also in line with what observed in the multitask setting with variance regularization (McDonald et al., 2016). The classification setting has not been used before and has been created ad-hoc for our purpose. In this case we have an increased variance probably due to the datasets being highly unbalanced. In order to investigate the impact of passing through the data only once in the different steps in our method, we conducted additional experiments. The results, presented in App. J, indicate that the single pass strategy is competitive with respect to the more expensive ERM.

7. Conclusion and Future Work

We have studied the performance of Stochastic Gradient Descent on the true risk regularized by the square euclidean distance to a bias vector, over a class of tasks. Drawing upon a learning-to-learn framework, we have shown that, when the variance of the tasks is relatively small, the introduction of an appropriate bias vector may bring a substantial benefit in comparison to the standard unbiased version, corresponding to learning the tasks independently. Then, we have proposed an efficient online meta-learning algorithm to estimate this bias and we have theoretically shown that the bias returned by our method can bring a comparable benefit. In the future, it would be interesting to investigate other kinds of relatedness among the tasks and to extend our analysis to other classes of loss functions, as well as to a Hilbert space setting. Finally, another valuable research direction is to derive fully dependent bounds, in which the hyperparameters are self-tuned during the learning process, see e.g. (Zhuang et al., 2019).

References

Alquier, P., Mai, T. T., and Pontil, M. Regret bounds for lifelong learning. In *Proceedings of the 20th International Conference on Artificial Intelligence and Statistics*, volume 54 of *Proceedings of Machine Learning Research*, pp. 261–269, 2017.

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

Bauschke, H. H. and Combettes, P. L. *Convex Analysis and Monotone Operator theory in Hilbert Spaces*, volume 408. Springer, 2011.

Baxter, J. A model of inductive bias learning. *J. Artif. Intell. Res.*, 12(149–198):3, 2000.

Beck, A. and Teboulle, M. A fast iterative shrinkage-thresholding algorithm for linear inverse problems. *SIAM journal on imaging sciences*, 2(1):183–202, 2009.

Borwein, J. and Zhu, Q. Techniques of variational analysis, ser, 2005.

Bousquet, O. and Elisseeff, A. Stability and generalization. *Journal of machine learning research*, 2(Mar):499–526, 2002.

Cesa-Bianchi, N., Conconi, A., and Gentile, C. On the generalization ability of on-line learning algorithms. *IEEE Transactions on Information Theory*, 50(9):2050–2057, 2004.

Denevi, G., Ciliberto, C., Stamos, D., and Pontil, M. Incremental learning-to-learn with statistical guarantees. In *Proc. 34th Conference on Uncertainty in Artificial Intelligence (UAI)*, 2018a.

Denevi, G., Ciliberto, C., Stamos, D., and Pontil, M. Learning to learn around a common mean. In *Advances in Neural Information Processing Systems*, pp. 10190–10200, 2018b.

Finn, C., Abbeel, P., and Levine, S. Model-agnostic meta-learning for fast adaptation of deep networks. In *Proceedings of the 34th International Conference on Machine Learning*, volume 70 of *Proceedings of Machine Learning Research*, pp. 1126–1135. PMLR, 2017.

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

Franceschi, L., Frasconi, P., Salzo, S., Grazzi, R., and Pontil, M. Bilevel programming for hyperparameter optimization and meta-learning. In *International Conference on Machine Learning*, PMLR 80, pp. 568–1577, 2018.

Gupta, R. and Roughgarden, T. A pac approach to application-specific algorithm selection. *SIAM Journal on Computing*, 46(3):992–1017, 2017.

Hazan, E. Introduction to online convex optimization. *Foundations and Trends in Optimization*, 2016.

Jean-Baptiste, H.-U. *Convex analysis and minimization algorithms: advanced theory and bundle methods*. SPRINGER, 2010.

Khodak, M., Balcan, M.-F., and Talwalkar, A. Provable guarantees for gradient-based meta-learning. *arXiv preprint arXiv:1902.10644*, 2019.

Kuzborskij, I. and Orabona, F. Fast rates by transferring from auxiliary hypotheses. *Machine Learning*, 106(2): 171–195, 2017.

Lacoste-Julien, S., Schmidt, M., and Bach, F. A simpler approach to obtaining an o (1/t) convergence rate for the projected stochastic subgradient method. *arXiv preprint arXiv:1212.2002*, 2012.

Lenk, P. J., DeSarbo, W. S., Green, P. E., and Young, M. R. Hierarchical bayes conjoint analysis: Recovery of part-worth heterogeneity from reduced experimental designs. *Marketing Science*, 15(2):173–191, 1996.

Maurer, A. Algorithmic stability and meta-learning. *Journal of Machine Learning Research*, 6:967–994, 2005.

Maurer, A. Transfer bounds for linear feature learning. *Machine Learning*, 75(3):327–350, 2009.

Maurer, A., Pontil, M., and Romera-Paredes, B. Sparse coding for multitask and transfer learning. In *International Conference on Machine Learning*, 2013.

Maurer, A., Pontil, M., and Romera-Paredes, B. The benefit of multitask representation learning. *The Journal of Machine Learning Research*, 17(1):2853–2884, 2016.

McDonald, A. M., Pontil, M., and Stamos, D. New perspectives on k-support and cluster norms. *Journal of Machine Learning Research*, 17(155):1–38, 2016.

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

Pentina, A. and Urner, R. Lifelong learning with weighted majority votes. In *Advances in Neural Information Processing Systems*, pp. 3612–3620, 2016.

Rakhlin, A., Shamir, O., Sridharan, K., et al. Making gradient descent optimal for strongly convex stochastic optimization. In *ICML*, volume 12, pp. 1571–1578. Citeseer, 2012.

Ravi, S. and Larochelle, H. Optimization as a model for few-shot learning. In *I5th International Conference on Learning Representations*, 2017.

Shalev-Shwartz, S. and Ben-David, S. *Understanding Machine Learning: From Theory to Algorithms*. Cambridge University Press, 2014.

Shalev-Shwartz, S. and Kakade, S. M. Mind the duality gap: Logarithmic regret algorithms for online optimization. In *Advances in Neural Information Processing Systems*, pp. 1457–1464, 2009.

Shamir, O. and Zhang, T. Stochastic gradient descent for non-smooth optimization: Convergence results and optimal averaging schemes. In *International Conference on Machine Learning*, pp. 71–79, 2013.

Thrun, S. and Pratt, L. *Learning to Learn*. Springer, 1998.

Zhuang, Z., Cutkosky, A., and Orabona, F. Surrogate losses for online learning of stepsizes in stochastic non-convex optimization. *arXiv preprint arXiv:1901.09068*, 2019.