Learning to Generalize Provably in Learning to Optimize

Junjie Yang[*] The Ohio State University
Tianlong Chen[*] UT Austin
Mingkang Zhu[*] UT Austin
Fengxiang He JD Explore Academy
Dacheng Tao JD Explore Academy
Yingbin Liang The Ohio State University
Zhangyang Wang UT Austin

Abstract

Learning to optimize (L2O) has gained increasing popularity, which automates the design of optimizers by data-driven approaches. However, current L2O methods often suffer from poor generalization performance in at least two folds: (i) applying the L2O-learned optimizer to unseen optimizees, in terms of lowering their loss function values *(optimizer generalization, or "generalizable learning of optimizers")*; and (ii) the test performance of an optimizee (itself as a machine learning model), trained by the optimizer, in terms of the accuracy over unseen data *(optimizee generalization, or "learning to generalize")*. While the optimizer generalization has been recently studied, the optimizee generalization (or learning to generalize) has not been rigorously studied in the L2O context, which is the aim of this paper. We first theoretically establish an implicit connection between the local entropy and the Hessian, and hence unify their roles in the handcrafted design of generalizable optimizers as equivalent metrics of the landscape flatness of loss functions. We then propose to incorporate these two metrics as *flatness-aware* regularizers into the L2O framework in order to meta-train optimizers to learn to generalize, and theoretically show that such generalization ability can be learned during the L2O meta-training process and then transformed to the optimizee loss function. Extensive experiments consistently validate the effectiveness of our proposals with substantially improved generalization on multiple sophisticated L2O models and diverse optimizees. Our code is available at: `https://github.com/VITA-Group/Open-L2O/tree/main/Model_Free_L2O/L2O-Entropy`.

Proceedings of the 26[th] International Conference on Artificial Intelligence and Statistics (AISTATS) 2023, Valencia, Spain. PMLR: Volume 206.

1 Introduction

One cornerstone of deep learning's success is the stochastic gradient-based optimization methods, such as SGD (Robbins and Monro, 1951), Adam (Kingma and Ba, 2014), AdaGrad (Duchi et al., 2011), RProp (Riedmiller and Braun, 1993), and RMSProp (Tieleman and Hinton, 2012). The performance of deep neural networks (DNNs) hinges on the choice of optimization methods and the corresponding parameter settings. Thus, intensive human labor is often required to empirically select the best optimization method and its parameters for each specific problem.

A promising data-driven approach, *learning to optimize (L2O)*, arises from the meta-learning community to alleviate this issue (Chen et al., 2022). It aims to replace traditional optimization algorithms (i.e., **optimizers**) tuned by humans, with optimizers parameterized by neural networks that can be trained to learn update rules from data. Existing works have demonstrated that such learned optimizers are able to decrease the objective function faster while tremendously reducing the required human labor. Andrychowicz et al. (2016a) first proposed to parameterize the update rules using a long short-term memory (LSTM) network. The LSTM optimizer tries to simulate the behavior of iterative methods by unrolling. By aggregating a set of loss functions (i.e., **optimizees**) to be optimized at each time step, it aims to minimize the overall loss along the optimization path. Wichrowska et al. (2017) enlarged the optimizer model to a hierarchical recurrent neural network (RNN) to improve its capability on larger or unseen optimization problems. Li and Malik (2016) also proposed a reinforcement learning based L2O approach.

All existing L2O methods so far aim at the goal of "learn-

[*]The first three authors have made equal contributions.

ing to optimize", i.e., a meta-learned optimizer can minimize a given optimizee loss function successfully. However, the generalization abilities, one of the core problems in machine learning, have not been explored thoroughly for current L2O methods. Specifically, there exist two different generalization concepts in the L2O context: **optimizer generalization** (or "generalizable learning of optimizers") and **optimizee generalization** (or "learning to generalize") (see Figure 1 for the difference between the two). *Optimizer generalization* characterizes how an optimizer trained by a certain set of given optimizees generalizes to unseen optimizees in terms of the unseen optimizee's training loss. On the other hand, *optimizee generalization* characterizes how an optimizee solution such as a classifier (trained by an optimizer) generalizes over the optimizee's unseen testing data. While the *optimizer generalization* has been recently studied in Almeida et al. (2021), the *optimizee generalization* has not been rigorously studied in the L2O context, which is the aim of this paper.

1.1 Main Contributions

This paper first examines the existing hand-crafted optimizer designs and provides an unified understanding of the two core metrics used for facilitating the generalization ability. We then propose the "learn to generalize" design, so that L2O can meta-train optimizers to have such generalization ability when they are applied to optimizees.

In the traditional design of generalizable optimizers, the metrics of **Hessian** (Keskar et al., 2017) and **local entropy** (Chaudhari et al., 2017) are often adopted to directly design optimizee loss functions in order to achieve good generalization. While *Hessian* directly measures the flatness of the loss landscape and facilitates the solution to a *flat basin*, the connection of *local entropy* to the loss geometry is rather implicit and has not been well understood.

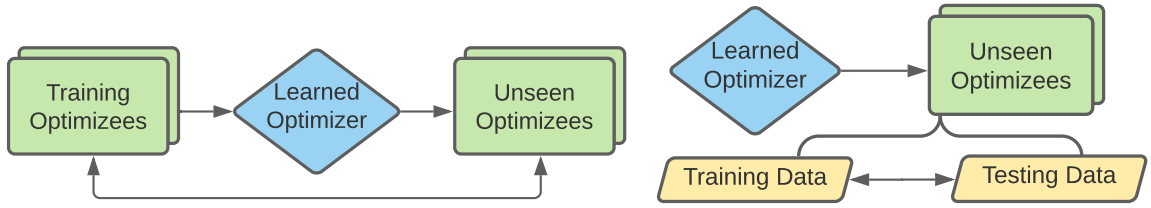
- The first contribution of the paper lies in establishing the implicit connection of local entropy to Hessian. Our theory explains that the existing Hessian and local entropy-based approaches are rooted in the same reason to improve the generalization performance. Specifically, we show theoretically that Hessian is upper bounded by a monotonically increasing function of the negative local entropy, and hence, large local entropy necessarily implies small Hessian. This explains that the Entropy-SGD algorithm (Chaudhari et al., 2017), by minimizing the negative local entropy-based loss function, facilitates a model solution with small Hessian and hence a flat landscape.

We then focus on the L2O problem and design a meta-training method for optimizers to "learn to generalize".

- The second contribution of the paper lies in proposing to use the *flatness-aware* regularizers based on Hessian and local entropy in the training of L2O optimizers. We show theoretically that such *flatness-aware* regularizers in L2O can meta-train optimizers to have good generalization abilities, i.e., such trained optimizers will favor the convergence to flat landscape of the loss functions and hence enhance the generalization ability of their trained optimizees, even when the optimizees do not have a generalization-based design. Our theory shows that the generalization ability can be learned during the meta-training process and transformed to the optimizee.
- We further provide comprehensive experiments over various tasks to demonstrate that our methods significantly improve the optimizee generalization ability of existing L2O methods, enabling them to outperform current state-of-the-art by a large margin. Our results also demonstrate that Hessian and local entropy yield very different practical performances. Local entropy is preferred when we adopt L2O to train large neural networks because it captures neighborhood landscape information which exhibits advantages in large neural networks. Instead, Hessian is preferred in small neural networks because it requires less time to compute.

2 Related Work

Learning to Optimize (L2O) As a special case of learning to learn, L2O has been widely investigated in various machine learning problems (Chen et al., 2017; Cao et al., 2019; Shen et al., 2021; Li et al., 2020; Chen et al., 2020b; Jiang et al., 2018; Xiong and Hsieh, 2020; You et al., 2020; Chen et al., 2020c; Metz et al., 2020; Merchant et al., 2021). The first L2O framework dates back to Andrychowicz et al. (2016b), in which the gradients and update rules of optimizee are formulated as the input features and outputs for an RNN optimizer, respectively. Later on, Li and Malik (2016) proposes an alternative reinforcement learning framework for L2O, leveraging gradient history and objective values as observations and step vectors as actions. Recently, more advanced variants arise to power up the generalization ability of L2O. For example, (i) regularizers such as random scaling, objective convexifying (Lv et al., 2017), and Jacobian regularization (Li et al., 2020), (ii) enhanced L2O model such as hierarchical RNN architecture (Wichrowska et al., 2017), and (iii) improved training techniques such as curriculum learning and imitation learning (Chen et al., 2020a). Moreover, Metz et al. (2021) introduces randomly initialized optimizers to form a positive feedback loop for effective training. Metz et al. (2019) proposes a training scheme that dynamically weights two unbiased gradient estimators for a variational loss, and overcomes the strongly bias and exploding norm restrictions in L2O. Differently from the previous optimizee regularizer design, our proposed *flatness-aware* regularizers are adopted to meta-train optimizers with good



(a) Optimizer Generalization (b) Optimizee Generalization

Figure 1: (a) Optimizer generalization characterizes the performance gap (training loss) between seen optimizees (during meta-training) and unseen optimizees (during meta-testing) by the same optimizer. (b) Optimizee generalization characterizes the performance gap between seen training data and unseen testing data, of the same unseen optimizee trained by an optimizer.

generalization abilities, even when the optimizees do not have generalization-based design.

Flatness on Generalization of Neural Networks Generalization analysis of neural networks has been widely studied by various methods, including VC-dimension (Bartlett et al., 2019), covering number (Bartlett et al., 2017), stability (Hardt et al., 2016; Zhou et al., 2018a), Rademacher complexity (Golowich et al., 2018; Ji and Liang, 2018; Ji et al., 2021; Arora et al., 2018, 2019), etc. In particular, the landscape *flatness* has been known to be associated with better generalization. Dinh et al. (2017) showed theoretically that sharp minimum can also generalize well for deep neural networks, but such a result does not contradict the fact that flat minima generalize well, which has strong evidence (He et al., 2019; Keskar et al., 2017). On the empirical side, Keskar et al. (2017) and He et al. (2019) showed that minima in wide valleys with small Hessian often generalize better than those in sharp basins with large Hessian. Further, Wilson et al. (2017) and Keskar and Socher (2017) showed empirically that SGD favors better generalization solutions than Adam. On the theory side, Zhou et al. (2020) showed that SGD is more unstable at sharp minima than Adam and Zou et al. (2021) explained that the inferior generalization performance of Adam is connected to nonconvex loss landscape. To improve the generalization performance, Entropy-SGD was introduced in Chaudhari et al. (2017) which was shown to outperform SGD in terms of the generalization error and the training time. Meanwhile, spectral norm regularization has been proposed in Yoshida and Miyato (2017) to improve the generalization ability of neural networks empirically. Further, Foret et al. (2021) proposed the SAM method, which minimizes the loss value and the loss sharpness simultaneously. In this paper, we further explain the good generalization of Entropy-SGD by connecting the local entropy to the Hessian. We then further show that both Hessian and local entropy can serve as good regularizers to train L2O optimizers, which can yield optimizees with good generalization.

Learning to Generalize Learning to generalize usually refers to domain generalization and domain adaptation (Csurka, 2017). Specifically, the goal of domain generalization is to learn a model which can generalize to unseen distributions and perform uniformly well across different data distributions (Carlucci et al., 2019; Dou et al., 2019; Li et al., 2018). Instead, this paper considers learning to generalize in the L2O context, which refers to the test performance of an optimizee and we call it *optimizee generalization*. Another related but different generalization notation in L2O is *optimizer generalization* or "generalizable learning of optimizers", which was recently studied in Almeida et al. (2021). Such optimizer generalization characterizes the training loss of unseen optimizees when we apply the L2O-learned optimizer.

3 Local Entropy and Generalization of Entropy-SGD

The local entropy was introduced in Chaudhari et al. (2017) as the performance metric of the landscape of the loss function. Specifically, let $L(\theta)$ be a loss function, and define the **local entropy** function of it as

$$G(\theta;\gamma) = \log \int_{\theta'} \exp\left(-L(\theta') - \frac{\gamma}{2}\|\theta-\theta'\|^2\right) \mathrm{d}\theta'.$$

Due to the exponential decay with respect to $\|\theta-\theta'\|^2$, the integral mainly captures the value of the loss function $L(\theta')$ over the neighborhood of θ. The local entropy has been applied in Chaudhari et al. (2017) to design Entropy-SGD, and the authors of Chaudhari et al. (2017) demonstrated that Entropy-SGD enjoys better Lipschitz and smoothness conditions while favoring better generalization solutions empirically. However, it is still not well understood what type of geometry the local entropy measures and why it facilitates improving the generalization performance.

Our following theorem establishes the implicit connection between the local entropy and Hessian and thus explains that the local entropy also measures the flatness of the loss function. In this way, minimization of the local entropy yields models in the flat landscape and hence with good generalization performance.

Theorem 1 (Connection between local entropy and Hessian)**.** *Consider a non-negative convex loss function $L(\theta)$, where $\theta \in \mathbb{R}^p$, and suppose Assumption 1 in Appendix B.1 holds. Then,*

$$\|\nabla^2 L(\theta)\| \leq D^{-1}(-G(\theta;\gamma)),$$

where $D^{-1}(x)$ is a monotonically increasing function, defined as the inverse function of $D(x) = \log(x+\gamma)+L(\theta)+(p-1)\log\gamma - mM - \frac{p}{2}\log(2\pi) - \frac{1}{2}\rho m^3 - C(\gamma,p,m)$, $C(\gamma,p,m) = \log\int_{\theta':\|\theta'-\theta\|>m}\exp(-\frac{\gamma}{2}\|\theta-\theta'\|^2)d\theta'$, $\theta \in \mathbb{R}^p$, m, M and ρ are constants.

It can be easily observed that the function $D(x)$ defined in Theorem 1 is monotonically increasing w.r.t. x, as determined by the only x-dependent term $\log(x+\gamma)$, and hence its inverse $D^{-1}(x)$ is also monotonically increasing.

Theorem 1 shows that small negative local entropy $-G(\theta;\gamma)$ implies small Hessian $\|\nabla^2 L(\theta)\|$. This explains that the Entropy-SGD algorithm, by minimizing a negative local entropy based loss function, facilitates a model solution with small Hessian and hence a flat landscape.

Note that Chaudhari et al. (2017) proposed the Entropy-SGD method based on local entropy, and showed that the local entropy loss function enjoys better Lipschitz and smoothness conditions and hence favors better generalization solutions. As a comparison, we here establish the connection of local entropy to Hessian, and hence explain its better generalization via its landscape flatness.

4 Learning to Generalize in L2O

In this section, we provide basic notations and our design of "learning to generalize" in L2O.

4.1 Preliminary

We define $l_{tr}(\theta;\xi)$ as the non-negative meta-training functions, where $\theta \in \mathbb{R}^p$ is the **optimizee** parameter, and ξ denotes training data samples. Suppose there are N training data samples $\xi \in \{\xi_i, i = (1,\ldots,N)\}$. Then we define the empirical meta-training function and its corresponding population risk function as follows:

$$\hat{L}_{tr}(\theta) = \frac{1}{N}\sum_{i=1}^{N} l_{tr}(\theta;\xi_i), \;\; L_{tr}(\theta) = \mathbb{E}_\xi l_{tr}(\theta;\xi). \quad (1)$$

An L2O algorithm aims to learn an *update rule* for optimizee θ based on the meta-training function. An update rule can be expressed as $\theta_{tr}^{t+1}(\phi) = \theta_{tr}^t(\phi) + m(z_{tr}^t;\phi)$, where $t = 0,1,\ldots,T-1$ denotes the iteration index over one epoch, the variable z captures the information (e.g., loss values, gradients) that we collect on the optimization path, and the **optimizer** function $m(z;\phi)$ is parameterized by ϕ and captures how the update of the **optimizee** parameter θ depends on the loss landscape information included in z. In order to find a desirable optimizer parameter ϕ, L2O solves the following meta-training problem:

$$\min_\phi \{\hat{L}_{tr}(\theta_{tr}^T(\phi))\}$$
$$\text{where} \quad \theta_{tr}^{t+1}(\phi) = \theta_{tr}^t(\phi) + m(z_{tr}^t;\phi). \quad (2)$$

A popular L2O meta-training algorithm applies the gradient descent method, which updates ϕ based on the gradient of the objective function $\hat{L}_{tr}(\theta_{tr}^T(\phi))$ with respect to ϕ. As suggested by eq. (2), each update of ϕ requires T iterations of the optimizee parameter $\theta_{tr}^0(\phi)$ to obtain $\theta_{tr}^T(\phi)$.

4.2 L2O Training via Flatness-aware Regularizers

In order to train optimizers with generalization ability, we propose to incorporate *flatness-aware* regularizers into L2O meta-training, so that such trained optimizers can learn to land the optimizee into a flat region, even if there is no generalization design for optimizee loss functions. We note that such an idea is fundamentally different from the existing handcrafted approaches, which directly design the optimizee loss function to feature flat landscapes. Rather, here we aim to let L2O auto-train optimizers to have generalization ability during meta-training, so that such optimizers will likely yield generalizable solutions when they are applied to optimizee loss functions, even when the optimizee function does not feature any flatness-aware design. We will show theoretically, such a "learning to generalize" design will guarantee the transformation of generalization performance from meta-training to optimizee loss functions. Our result will also characterize the impact of the similarity between training and testing tasks as well as the difference between the training and testing losses on the generalization performance.

The first regularizer we introduce is based on the spectral norm of the *Hessian*, smaller values of which correspond to a flatter landscape. Thus, the new L2O meta-training objective is given by:

$$\min_\phi \{\hat{L}_{tr}(\theta_{tr}^T(\phi)) + \lambda\|\nabla_\theta^2 \hat{L}_{tr}(\theta_{tr}^T(\phi))\|\}$$
$$\text{where} \quad \theta_{tr}^{t+1}(\phi) = \theta_{tr}^t(\phi) + m(z_{tr}^t;\phi), \quad (3)$$

where λ is the regularizer hyperparameter. Note that the Hessian regularizer is adopted for training the optimizer parameter ϕ, and its impact on the update rule $m(z_{tr}^t;\phi)$ is only through ϕ, i.e., the information in z^t does not include such regularization. Due to the computational intractability of directly penalizing $\nabla_\theta^2 \hat{L}_{tr}(\theta_{tr}^T(\phi))$, we investigate three approximation variants in the implementation. The details can be found in Section 5.3.

The second *flatness-aware* regularizer we incorporate to L2O is based on the local entropy function defined

in Section 3. Specifically, consider the loss function $\hat{L}_{tr}(\theta)$. Its local entropy is given by $\hat{G}_{tr}(\theta;\gamma) = \log \int_{\theta'} \exp\left(-\hat{L}_{tr}(\theta') - \frac{\gamma}{2}\|\theta-\theta'\|^2\right) \mathrm{d}\theta'$. Due to Theorem 1, the value of $\hat{G}_{tr}(\theta;\gamma)$ measures the flatness of the local area around θ. Thus, the L2O meta-training objective with the local entropy regularizer is given by:

$$\min_{\phi}\{\hat{L}_{tr}(\theta_{tr}^T(\phi)) - \lambda \hat{G}_{tr}(\theta_{tr}^T(\phi);\gamma)\}$$
$$\text{where} \quad \theta_{tr}^{t+1}(\phi) = \theta_{tr}^t(\phi) + m(z_{tr}^t;\phi). \quad (4)$$

In order to implement the gradient descent algorithm for meta-training, the gradient $-\nabla_\phi \hat{G}_{tr}(\theta_{tr}^T(\phi);\gamma)$ can be calculated by the entropy gradient $-\nabla_\theta \hat{G}_{tr}(\theta;\gamma)$ and the chain rule. In particular, as given in Chaudhari et al. (2017), the entropy gradient takes the following form

$$-\nabla_\theta \hat{G}_{tr}(\theta;\gamma) = \gamma(\theta - \mathbb{E}[\theta';\xi]), \quad (5)$$

where $\xi \in \{\xi_i, i = (1,\ldots,N)\}$ are training samples and the distribution of θ' is given by $P(\theta';\theta,\gamma) \propto \exp\left[-\hat{L}_{tr}(\theta') - \frac{\gamma}{2}\|\theta-\theta'\|^2\right]$.

4.3 Generalization Guarantee for Regularized L2O

To analyze the optimizee generalization abilities, we first define $l_{ts}(\theta;\zeta)$ as the non-negative meta-testing function, where $\theta \in \mathbb{R}^p$ is the **optimizee** parameter, and ζ denotes testing data samples. Suppose there are M testing data samples $\zeta \in \{\zeta_i, i = (1,\ldots,M)\}$. Then we define the empirical meta-testing function and its corresponding population risk function as follows:

$$\hat{L}_{ts}(\theta) = \frac{1}{M}\sum_{j=1}^{M} l_{ts}(\theta;\zeta_j), \;\; L_{ts}(\theta) = \mathbb{E}_\zeta l_{ts}(\theta;\zeta). \quad (6)$$

In meta-testing, we apply the output ϕ of meta-training and its corresponding optimizer to update the optimizee as $\theta_{ts}^{t+1}(\phi) = \theta_{ts}^t(\phi) + m(z_{ts}^t;\phi)(t = 0,1,\ldots,T-1)$. Note that we differentiate the optimizee updates in training and testing by subscripts tr and ts, respectively.

Furthermore, we let ϕ^* be the optimal optimizer parameter for Hessian-regularized L2O, which can be written as

$$\phi^* = \arg\min_{\phi}\{\hat{L}_{tr}(\theta_{tr}^T(\phi)) + \lambda\|\nabla_\theta^2 \hat{L}_{tr}(\theta_{tr}^T(\phi))\|\}. \quad (7)$$

Motivated by optimization theory, we note that the regularized optimization problem in eq. (7) is equivalent to the following constrained optimization

$$\min_{\phi}\{\hat{L}_{tr}(\theta_{tr}^T(\phi))\} \quad \text{where } \theta_{tr}^{t+1}(\phi) = \theta_{tr}^t(\phi) + m(z_{tr}^t;\phi)$$
$$\text{subject to } \|\nabla_\theta^2 \hat{L}_{tr}(\theta_{tr}^T(\phi))\| \le B_{\text{Hessian}}(\lambda), \quad (8)$$

where $B_{\text{Hessian}}(\lambda)$ is the constraint bound on the Hessian determined by λ. Thus, the optimizer parameter ϕ^* learned by the Hessian-regularized L2O meta-training in eq. (7) is also a solution to eq. (8), i.e., its Hessian satisfies the constraint. Then we let $\theta_{ts}^T(\phi^*)$ denote the optimizee parameters trained by optimizer ϕ^* in meta-testing, and θ_{ts}^* denote the optimal point of the population meta-testing function $L_{ts}(\cdot)$. We then characterize the generalization error as $L_{ts}(\theta_{ts}^T(\phi^*)) - L_{ts}(\theta_{ts}^*)$, which captures how well the optimizer ϕ^* performs on a testing task with respect to the best possible testing loss value.

The following theorem characterizes the generalization performance of the optimizee trained with Hessian regularized optimizer as defined above.

Theorem 2 (Generalization Error of Hessian-Regularized L2O)**.** *Suppose Assumptions 1, 2, 3, 4 in Appendix B.1 and C.1 hold. We let* $N \ge \max\{4Cp\log N/\eta_*^2, Cp\log p\}$ *where* $C = C_0\max\{c_h, 1, \log(\frac{r\tau}{\delta})\}$, $\eta_*^2 = \min\{\frac{\epsilon^2}{\tau^2}, \frac{\eta^2}{\tau^4}, \frac{\eta^4}{\rho^2\tau^2}\}$ *and* C_0 *is a universal constant. Then, with probability at least* $1-2\delta$*, we have*

$$L_{ts}(\theta_{ts}^T(\phi^*)) - L_{ts}(\theta_{ts}^*)$$
$$\le \tfrac{1}{2}\Big(\Delta_T^* + \Delta_\theta^* + \mathcal{O}(w^{T-T'}) + \mathcal{O}(\sqrt{\tfrac{C\log N}{N}})\Big)^2$$
$$\Big(B_{Hessian}(\lambda) + \Delta_1^* + \mathcal{O}(w^{T-T'}) + \mathcal{O}(\sqrt{\tfrac{C\log N}{N}})\Big),$$

where $\Delta_T^* = \|\theta_{ts}^T(\phi^*) - \theta_{tr}^T(\phi^*)\|$, $\Delta_\theta^* = \|\theta_{tr}^* - \theta_{ts}^*\|$, $\Delta_H^* = \|\nabla_\theta^2 L_{ts}(\theta_{ts}^*) - \nabla_\theta^2 L_{tr}(\theta_{tr}^*)\|$, $\Delta_1^* = \rho\Delta_T^* + \rho\Delta_\theta^* + \Delta_H^*$, $w = \frac{L-\mu}{L+\mu}$, T' *is the minimum gradient descent iterations for* $\theta_{tr}^{T'}(GD)$ *to enter into the local basin of* $\theta_{tr}^T(\phi^*)$ *and GD refers to Gradient Descent.*

Theorem 2 characterizes the impact of the Hessian regularizer on the generalization error by the term $B_{\text{Hessian}}(\lambda)$. Clearly, by choosing the regularization hyperparameter λ, we control the value of $B_{\text{Hessian}}(\lambda)$ and further the generalization error. Specifically, larger λ corresponds to smaller $B_{\text{Hessian}}(\lambda)$ and hence yields a smaller generalization error. This also explains that flatter landscape (i.e., smaller $B_{\text{Hessian}}(\lambda)$ on Hessian) yields better generalization performance (i.e., smaller generalization error).

The generalization error in Theorem 2 also contains other terms which we explain as follows: (a) $\Delta_1^* = \rho\Delta_T^* + \rho\Delta_\theta^* + \Delta_H^*$ captures the similarities between the training and testing tasks: more similar tasks yield better generalization. These errors are owing to transformations of the generalization design in meta-training to optimizee training and testing; (b) $\mathcal{O}(w^{T-T'})$ captures the exponential decay rate of the optimizee's iteration due to the strong convexity, and vanishes for large T; and (c) $\mathcal{O}(\sqrt{\frac{C\log N}{N}})$ arises due to the differences between the empirical and population loss functions, and vanishes as the sample size N gets large.

We next analyze the generalization error of the Entropy regularizer on the optimizee generalization ability. Similarly

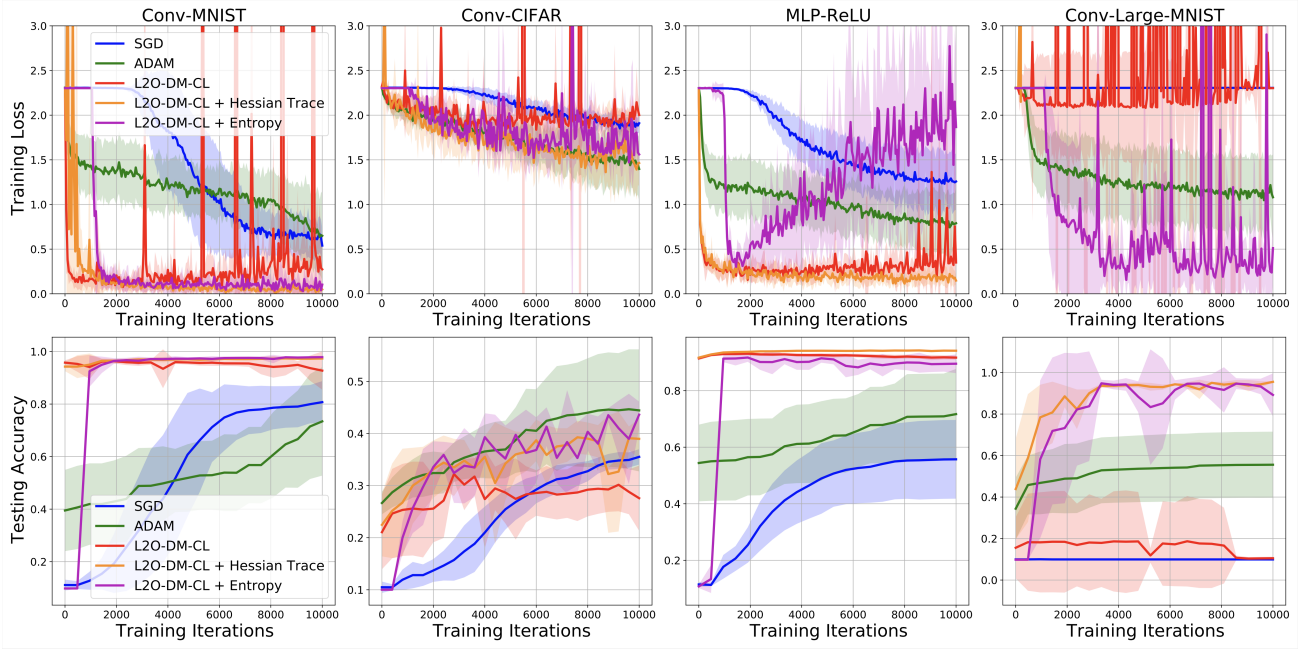


Figure 2: Comparison of the training loss/testing accuracy of optimizees trained using analytical optimizers and L2O-DM-CL (Chen et al., 2020a) with/without the proposed Hessian/Entropy regularization.

to the Hessian regularizer, we let ϕ^* be the optimal optimizer parameter, which can be written as :

$$\phi^* = \arg\min_{\phi} \{\hat{L}_{tr}(\theta_{tr}^T(\phi)) - \lambda \hat{G}_{tr}(\theta_{tr}^T(\phi); \gamma)\}. \quad (9)$$

Meanwhile, the regularized optimization problem in eq. (9) is equivalent to the following constrained optimization:

$$\min_{\phi} \{\hat{L}_{tr}(\theta_{tr}^T(\phi))\} \quad \text{where} \quad \theta_{tr}^{t+1} = \theta_{tr}^t + m(z_{tr}^t; \phi)$$
$$\text{subject to } -\hat{G}_{tr}(\theta_{tr}^T(\phi); \gamma) \leq B_{\text{Entropy}}(\lambda), \quad (10)$$

where $B_{\text{Entropy}}(\lambda)$ is the constraint on the Entropy determined by λ. Thus, the optimizer ϕ^* learned by the Entropy-regularized L2O meta-training in eq. (9) is also a solution to eq. (10), i.e., the local entropy satisfies the constraint.

Corollary 1 (Generalization Error of Entropy-Regularized L2O)**.** *Suppose the same conditions of Theorem 2 hold. Then the generalization error of L2O with Entropy regularizer takes the bound in Theorem 2 with $B_{Hessian}(\lambda)$ being replaced by $D^{-1}(B_{Entropy}(\lambda))$.*

Corollary 1 shows that the bound $D^{-1}(B_{\text{Entropy}}(\lambda))$ serves the same role as the Hessian bound in the generalization performance. Thus, by controlling the hyperparameter λ to be large enough in the L2O training, $B_{\text{Entropy}}(\lambda)$ as well as $D^{-1}(B_{\text{Entropy}}(\lambda))$ and Hessian can be controlled to be sufficiently small. In this way, the optimizee will be landed into a flat basin to enjoy better generalization.

5 Experiments

We consider two L2O algorithms: L2O-DM-CL* (Chen et al., 2020a) and L2O-Scale (Wichrowska et al., 2017). For training L2O-Scale, we use a three-layer convolutional neural network (CNN) which has one fully-connected layer, and two convolutional layers with eight 3×3 and 5×5 kernels respectively. For training L2O-DM, we adopt the same meta training optimizee from (Andrychowicz et al., 2016b), which is a simple Multi-Layer Perceptron (MLP) with one hidden layer of 20 dimensions and the sigmoid activation. MNIST is used for all meta-training.

Meta Testing Optimizees. We select four distinct and representative meta testing optimizees from (Andrychowicz et al., 2016b) and (Chen et al., 2020a) to evaluate the generalization ability of the learned optimizer. Specifically, ① `MLP-ReLU`: a single layer MLP with 20 neurons and the ReLU activation function on MNIST. ② `Conv-MNIST`: a CNN has one fully-connected layer, and two convolutional layers with 16 3×3 and 32 5×5 kernels on MNIST. ③ `Conv-Large-MNIST`: a large CNN has one fully-connected layer, and four convolutional layers with two 32 3×3 and two 32 5×5 kernels on MNIST. ④ `Conv-CIFAR`: a CNN has one fully-connected layer, and two convolutional layers with 16 3×3 and 32 5×5 kernels on CIFAR-10 (Krizhevsky and Hinton, 2009). Optimizees ①, ②, and ③ are for evaluating the generalization of L2O across network architectures. Then, ④ evaluates the generalization of

*It is an enhanced version of the L2O-DM introduced by DeepMind Andrychowicz et al. (2016b). We choose it as a stronger baseline with improved generalization.

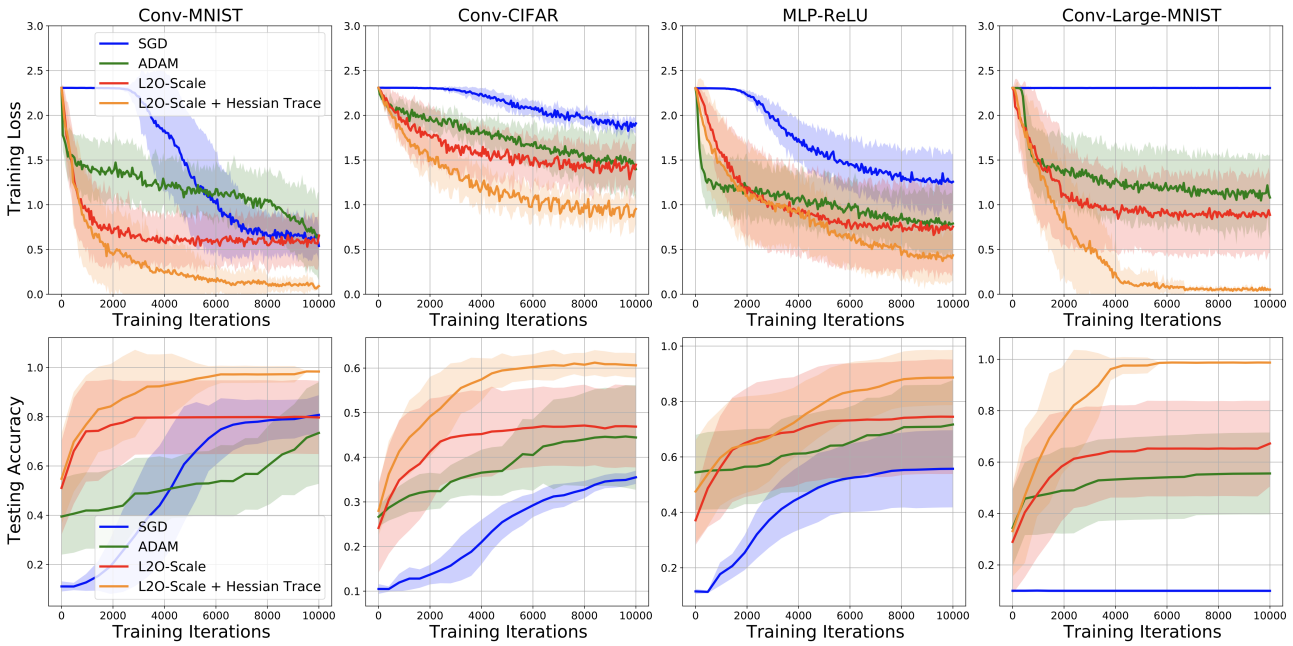


Figure 3: Comparison of the training loss/testing accuracy of optimizees trained using analytical optimizers and L2O-Scale (Wichrowska et al., 2017) with/without the proposed Hessian regularization.

L2O across both network architectures and datasets.

Training and Evaluation details. During the meta-training stage of L2O, L2O-Scale is trained with 5 epochs, where the number of each epoch's iteration is drawn from a heavy-tailed distribution (Wichrowska et al., 2017). L2O-DM-CL is trained with a curriculum schedule of training epochs and iterations, following the default setup in Chen et al. (2020a). RMSprop with the learning rate 1×10^{-6} is used to update L2Os. For the {Hessian, Entropy} regularization coefficients $\{\lambda_{\text{Hessian}}, \lambda_{\text{Entropy}}, \gamma\}$, we perform a grid search and choose $\{5 \times 10^{-5}$,-,-$\}$/$\{1 \times 10^{-10}, 1 \times 10^{-6}, 1 \times 10^{-4}\}$ for L2O-Scale/L2O-DM-CL.

In the meta-testing stage of L2O, we compare our methods with classical optimizers like SGD and Adam, and state-of-the-art (SOTA) L2Os such as L2O-Scale and L2O-DM-CL. Hyperparameters of both classical optimizers and L2O baselines are carefully tuned by the grid search and all other irrelevant variables are strictly controlled for a fair comparison. We run $10,000$ iterations for the meta-testing, and the corresponding training loss and test accuracy on all **unseen** optimizees are collected to evaluate the *optimizer* and *optimizee generalization*. We conduct **ten** independent replicates with different random seeds and all experiments are conducted on NVIDIA GeForce GTX 1080Ti GPUs.

5.1 Learning to Generalize with Hessian Regularization

In this section, we conduct extensive evaluations of our proposed Hessian regularization on previous state-of-the-art L2O methods, i.e., L2O-Scale (Wichrowska et al., 2017) and L2O-DM-CL (Chen et al., 2020a). Achieved training loss and testing accuracy are collected in Figure 3 and 2 which also include comparisons with representative analytical optimizers like SGD (Ruder, 2016) and Adam (Kingma and Ba, 2014). Note that the training loss corresponds to $\hat{L}_{tr}$ and test accuracy corresponds to L_{ts}. Several consistent observations can be drawn from our results:

❶ Hessian Trace regularizer consistently enhances the generalization abilities of learned L2Os and trained optimizees. Specifically, L2Os with Hessian Trace enable fast training loss decay and much lower final loss on all four unseen meta-testing optimizees, showing the improved *optimizer generalization* ability, which is great byproduct of our regularizer. Furthermore, all unseen optimizees trained by Hessian regularized L2Os enjoy substantial testing accuracy which boosts up to 31%, demonstrating the enhanced *optimizee generalization* ability. Such impressive performance gains effective evidence of our proposal, which again suggests that Hessian regularization enables optimizers to learn to generalize.

❷ Adopting vanilla L2O-DM-CL to train meta-testing optimizees (e.g., `Conv-MNIST` and `Conv-CIFAR`) suffers from instability as shown in Figure 2, and it can be significantly mitigated by introducing our flatness-aware regularization. `Conv-Large-MNIST` is an exception, where the L2O-DM-CL fails to train this optimizee and ends up with random-guess accuracies, i.e., 10%. Although plugging Hessian Trace into L2O-DM-CL greatly improves its test accuracy from 10% to $95\%+$, it still undergoes an unsatisfactory training loss. Reasons may lie in the rough model architecture and limited input features of L2O-DM-CL, co-

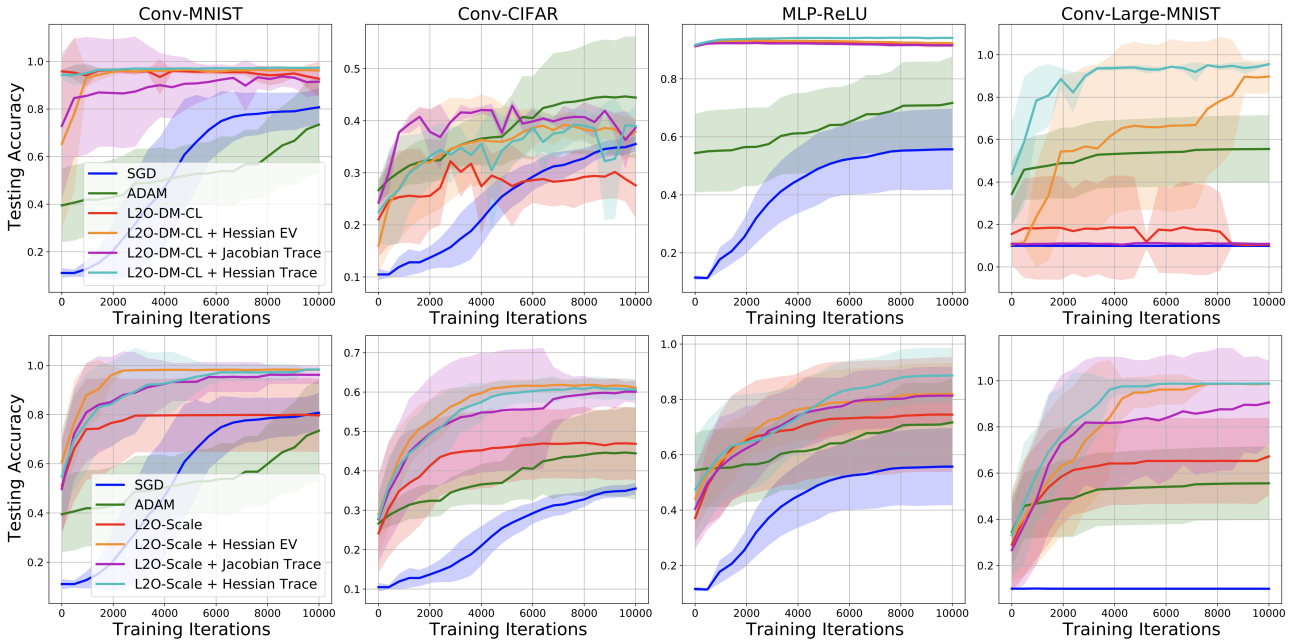


Figure 4: Comparison of the testing accuracy of optimizees trained using analytical optimizers and SOTA L2O with/without different Hessian regularization, *Hessian EV*, *Hessian Trace*, and *Jacobian Trace*.

inciding with the findings in (Chen et al., 2020a). We will investigate this interesting phenomenon in the future.

❸ For advanced L2O-Scale, Hessian Trace regularization facilitates it to converge to significantly lower minima and to obtain considerable accuracy improvements. It enlarges the advantages of L2O methods compared to analytical optimizers, SGD and Adam, unleashing the power of parameterized optimizers.

5.2 Learning to Generalize with Entropy Regularization

We investigate the generalization improvements from the Entropy regularization. Generally, it boosts optimizee generalization of L2O in most cases, as shown in Figure 2.

Hessian v.s. Entropy Regularization. We compare our two kinds of flatness-aware regularizers from both computational cost and performance benefits perspectives.

❶ In order to calculate the local entropy's gradient in eq. (5), it involves gradients from multiple unroll steps for the estimation (Chaudhari et al., 2017), leading to extra memory and computing outlays. Compared to Hessian augmented L2O, it costs $\sim$ 2.6x memory and $\sim$ 3x running time for L2O-DM-CL experiments*. Meanwhile, Hessian augmented L2O requires around 10% more memory cost compared with vanilla L2O since we approximate Hessian in practice. However, the wall clock comparison included in Appendix A.2 shows that these two regularizers share the same inference time which requires only $\sim$ 1.5x time than analytical optimizers. In comparison, another flatness-aware optimizer SAM (Foret et al., 2021), which incorporates loss landscape in the loss function, takes longer training time (1.5x $\sim$ 2x SGD).

❷ As for generalization gains, Entropy regularizer performs slightly better on `Conv-MNIST` and `Conv-CIFAR`, while behaves marginally worse on `MLP-ReLU` and `Conv-Large-MNIST` compared to Hessian regularizer. We would like to draw the reader's attention to `Conv-Large-MNIST`, in which Entropy regularized L2O-DM-CL is capable of decaying the training loss and finding a much lower minimum than Adam. Note that on this optimizee, both L2O-DM-CL and its Hessian variant can not decrease the training loss. The possible reason is that multi-layer convolutional neural networks without BN cannot be stably trained on MNIST. However, our L2O-DM-CL+Entropy is more stable in training and improves testing accuracy compared with L2O-DM-CL. This indicates that L2O-DM-CL + Entropy may also produce a more trainable loss surface for optimizees.

Based on the above experiments as well as those in Appendix A.1, we observe that L2O+Entropy is preferred when we adopt L2O to train large neural networks, where L2O+Entropy yields better optimizer and optimizee generalization abilities. Meanwhile, L2O+Hessain optimizer requires less time per iteration to train and achieves lower training loss as well as higher test accuracy than

*We conduct entropy-related experiments on light-weight L2O-DM-CL rather heavy L2O-Scale models, since RTX TITAN with 24G memory is the largest GPU we can access and afford.

L2O+Entropy in small MLPs. The possible reason is that Entropy takes account of the landscape over a large range of loss to measure the flatness, and can hence capture complex landscape information in large neural networks. On the other hand, the Hessian regularizer captures the flatness information only for the individual point, but in a more accurate manner, and thus is more suitable to smaller neural networks with a relatively simple landscape. We also compare our proposed methods with Entropy-SGD and SGD with Hessian regularization in Appendix A.3 which demonstrates meta-training's advantages.

5.3 Ablation and Visualization

In this section, we carefully examine the effect of Hessian regularization's different approximation variants, including ① *Hessian EV*: the eigenvalue of largest module of Hessian matrix, computed by power iteration (Yao et al., 2020); ② *Hessian Trace*: the trace of Hessian matrix, calculated via Hutchinson method (Yao et al., 2020); ③ *Jacobian Trace*: the trace of Hessian's Jacobian approximation $\nabla_\theta \hat{L}_{tr}(\theta_{tr}^T(\phi))^\top \nabla_\theta \hat{L}_{tr}(\theta_{tr}^T(\phi))$. Note that such Hessian approximation methods do not involve computing Hessian explicitly which helps to reduce the memory and computational cost and we adopt 10 iterations for Hessian norms' approximation. Results are presented in Figure 4. We find that the Hessian Trace regularizer achieves the most stable and substantial performance boosts across all optimizees. Jacobian Trace performs the worst which is within expectation since it provides the roughest estimation of Hessian.

6 Conclusion

In this paper, we first establish an implicit connection between the local entropy and the Hessian. Then we propose *flatness-aware* regularizers to incorporate these two metrics into the L2O framework for meta-training optimizers to learn to generalize. We further establish the theoretical guarantee to show that such generalization ability during L2O meta-training can be transformed to improve the optimizee's generalization over testing data. Our empirical results validate the effectiveness of our proposal, taking a further step for L2O usage in real-world scenarios.

Acknowledgements

The work of Y. Liang was supported in part by the U.S. National Science Foundation under the grant ECCS-2113860. The work of Z. Wang was supported in part by the U.S. National Science Foundation under the grant ECCS-2113904.

References

Almeida, D., Winter, C., Tang, J., and Zaremba, W. (2021). A generalizable approach to learning optimizers. *arXiv preprint arXiv:2106.00958*.

Andrychowicz, M., Denil, M., Gómez, S., Hoffman, M. W., Pfau, D., Schaul, T., Shillingford, B., and de Freitas, N. (2016a). Learning to learn by gradient descent by gradient descent. In Lee, D., Sugiyama, M., Luxburg, U., Guyon, I., and Garnett, R., editors, *Advances in Neural Information Processing Systems (NeurIPS)*, volume 29. Curran Associates, Inc.

Andrychowicz, M., Denil, M., Gomez, S., Hoffman, M. W., Pfau, D., Schaul, T., Shillingford, B., and De Freitas, N. (2016b). Learning to learn by gradient descent by gradient descent. In *Advances in neural information processing systems (NeurIPS)*.

Arora, S., Du, S., Hu, W., Li, Z., and Wang, R. (2019). Fine-grained analysis of optimization and generalization for overparameterized two-layer neural networks. In *International Conference on Machine Learning (ICML)*.

Arora, S., Ge, R., Neyshabur, B., and Zhang, Y. (2018). Stronger generalization bounds for deep nets via a compression approach. In *International Conference on Machine Learning (ICML)*.

Bartlett, P. L., Foster, D. J., and Telgarsky, M. J. (2017). Spectrally-normalized margin bounds for neural networks. In *Advances in Neural Information Processing Systems (NeurIPS)*.

Bartlett, P. L., Harvey, N., Liaw, C., and Mehrabian, A. (2019). Nearly-tight VC-dimension and pseudodimension bounds for piecewise linear neural networks. *The Journal of Machine Learning Research (JMLR)*.

Cao, Y., Chen, T., Wang, Z., and Shen, Y. (2019). Learning to optimize in swarms. In *Advances in Neural Information Processing Systems (NeurIPS)*, pages 15018–15028.

Carlucci, F. M., D'Innocente, A., Bucci, S., Caputo, B., and Tommasi, T. (2019). Domain generalization by solving jigsaw puzzles. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*.

Chaudhari, P., Choromanska, A., Soatto, S., LeCun, Y., Baldassi, C., Borgs, C., Chayes, J., Sagun, L., and Zecchina, R. (2017). Entropy-SGD: Biasing gradient descent into wide valleys. In *International Conference on Learning Representations (ICLR)*.

Chen, T., Chen, X., Chen, W., Heaton, H., Liu, J., Wang, Z., and Yin, W. (2022). Learning to optimize: A primer and a benchmark. *Journal of Machine Learning Research*, 23(189):1–59.

Chen, T., Zhang, W., Zhou, J., Chang, S., Liu, S., Amini, L., and Wang, Z. (2020a). Training stronger baselines for learning to optimize. *arXiv preprint arXiv:2010.09089*.

Chen, W., Yu, Z., Wang, Z., and Anandkumar, A. (2020b). Automated synthetic-to-real generalization. In *International Conference on Machine Learning (ICML)*, pages 1746–1756.

Chen, X., Chen, W., Chen, T., Yuan, Y., Gong, C., Chen, K., and Wang, Z. (2020c). Self-pu: Self boosted and calibrated positive-unlabeled training. In *International Conference on Machine Learning (ICML)*, pages 1510–1519.

Chen, Y., Hoffman, M. W., Colmenarejo, S. G., Denil, M., Lillicrap, T. P., Botvinick, M., and De Freitas, N. (2017). Learning to learn without gradient descent by gradient descent. In *International Conference on Machine Learning (ICML)*, pages 748–756.

Csurka, G. (2017). *Domain adaptation in computer vision applications*. Springer.

Dinh, L., Pascanu, R., Bengio, S., and Bengio, Y. (2017). Sharp minima can generalize for deep nets. In *International Conference on Machine Learning (ICML)*.

Dou, Q., Coelho de Castro, D., Kamnitsas, K., and Glocker, B. (2019). Domain generalization via model-agnostic learning of semantic features. *Advances in Neural Information Processing Systems (NeurIPS)*, 32.

Du, S. S., Zhai, X., Poczos, B., and Singh, A. (2019). Gradient descent provably optimizes over-parameterized neural networks. In *International Conference on Learning Representations (ICLR)*.

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

Foret, P., Kleiner, A., Mobahi, H., and Neyshabur, B. (2021). Sharpness-aware minimization for efficiently improving generalization. In *International Conference on Learning Representations (ICLR)*.

Golowich, N., Rakhlin, A., and Shamir, O. (2018). Size-independent sample complexity of neural networks. In *Conference On Learning Theory (COLT)*.

Hardt, M., Recht, B., and Singer, Y. (2016). Train faster, generalize better: Stability of stochastic gradient descent. In *International Conference on Machine Learning (ICML)*.

He, H., Huang, G., and Yuan, Y. (2019). Asymmetric valleys: Beyond sharp and flat local minima. In *Advances in Neural Information Processing Systems (NeurIPS)*, volume 32.

Ji, K. and Liang, Y. (2018). Minimax estimation of neural net distance. In *Advances in Neural Information Processing Systems (NeurIPS)*.

Ji, K., Zhou, Y., and Liang, Y. (2021). Understanding estimation and generalization error of generative adversarial networks. *IEEE Transactions on Information Theory*.

Jiang, H., Chen, Z., Shi, Y., Dai, B., and Zhao, T. (2018). Learning to defense by learning to attack. *arXiv preprint arXiv:1811.01213*.

Keskar, N. S., Nocedal, J., Tang, P. T. P., Mudigere, D., and Smelyanskiy, M. (2017). On large-batch training for deep learning: Generalization gap and sharp minima. In *International Conference on Learning Representations (ICLR)*.

Keskar, N. S. and Socher, R. (2017). Improving generalization performance by switching from Adam to SGD. *arXiv preprint arXiv:1712.07628*.

Kingma, D. P. and Ba, J. (2014). Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*.

Krizhevsky, A. and Hinton, G. (2009). Learning multiple layers of features from tiny images. *Master's thesis, Department of Computer Science, University of Toronto*.

Li, C., Chen, T., You, H., Wang, Z., and Lin, Y. (2020). Halo: Hardware-aware learning to optimize. In *European Conference on Computer Vision (ECCV)*, pages 500–518. Springer.

Li, D., Yang, Y., Song, Y.-Z., and Hospedales, T. M. (2018). Learning to generalize: Meta-learning for domain generalization. In *Thirty-Second AAAI Conference on Artificial Intelligence (AAAI)*.

Li, K. and Malik, J. (2016). Learning to optimize. *arXiv preprint arXiv:1606.01885*.

Li, Y. and Yuan, Y. (2017). Convergence analysis of two-layer neural networks with ReLU activation. *Advances in Neural Information Processing Systems (NeurIPS)*.

Lv, K., Jiang, S., and Li, J. (2017). Learning gradient descent: Better generalization and longer horizons. In *International Conference on Machine Learning (ICML)*, pages 2247–2255.

Mei, S., Bai, Y., and Montanari, A. (2018). The landscape of empirical risk for nonconvex losses. *The Annals of Statistics*.

Merchant, A., Metz, L., Schoenholz, S. S., and Cubuk, E. D. (2021). Learn2hop: Learned optimization on rough landscapes. In *International Conference on Machine Learning (ICML)*, pages 7643–7653.

Metz, L., Freeman, C. D., Maheswaranathan, N., and Sohl-Dickstein, J. (2021). Training learned optimizers with randomly initialized learned optimizers. *arXiv preprint arXiv:2101.07367*.

Metz, L., Maheswaranathan, N., Nixon, J., Freeman, D., and Sohl-Dickstein, J. (2019). Understanding and correcting pathologies in the training of learned optimizers. In *International Conference on Machine Learning (ICML)*, pages 4556–4565.

Metz, L., Maheswaranathan, N., Sun, R., Freeman, C. D., Poole, B., and Sohl-Dickstein, J. (2020). Using a thousand optimization tasks to learn hyperparameter search strategies. *arXiv preprint arXiv:2002.11887*.

Milne, T. (2019). Piecewise strong convexity of neural networks. In *Advances in Neural Information Processing Systems (NeurIPS)*, volume 32.

Riedmiller, M. and Braun, H. (1993). A direct adaptive method for faster backpropagation learning: the rprop algorithm. In *IEEE International Conference on Neural Networks*.

Robbins, H. and Monro, S. (1951). A stochastic approximation method. *The Annals of Mathematical Statistics*, 22(3):400–407.

Ruder, S. (2016). An overview of gradient descent optimization algorithms. *arXiv preprint arXiv:1609.04747*.

Safran, I. and Shamir, O. (2016). On the quality of the initial basin in overspecified neural networks. In *International Conference on Machine Learning (ICML)*.

Shen, J., Chen, X., Heaton, H., Chen, T., Liu, J., Yin, W., and Wang, Z. (2021). Learning a minimax optimizer: A pilot study. In *International Conference on Learning Representations (ICLR)*.

Tieleman, T. and Hinton, G. (2012). Lecture 6.5—RmsProp: Divide the gradient by a running average of its recent magnitude. COURSERA: Neural Networks for Machine Learning.

Wichrowska, O., Maheswaranathan, N., Hoffman, M. W., Colmenarejo, S. G., Denil, M., de Freitas, N., and Sohl-Dickstein, J. (2017). Learned optimizers that scale and generalize. In *International Conference on Machine Learning (ICML)*.

Wilson, A. C., Roelofs, R., Stern, M., Srebro, N., and Recht, B. (2017). The marginal value of adaptive gradient methods in machine learning. In *Advances in Neural Information Processing Systems (NeurIPS)*.

Xiong, Y. and Hsieh, C.-J. (2020). Improved adversarial training via learned optimizer.

Yao, Z., Gholami, A., Keutzer, K., and Mahoney, M. W. (2020). Pyhessian: Neural networks through the lens of the Hessian. In *2020 IEEE International Conference on Big Data (Big Data)*, pages 581–590. IEEE.

Yoshida, Y. and Miyato, T. (2017). Spectral norm regularization for improving the generalizability of deep learning. *arXiv preprint arXiv:1705.10941*.

You, Y., Chen, T., Wang, Z., and Shen, Y. (2020). L2-gcn: Layer-wise and learned efficient training of graph convolutional networks. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 2127–2135.

Zhou, P., Feng, J., Ma, C., Xiong, C., Hoi, S. C. H., et al. (2020). Towards theoretically understanding why SGD generalizes better than Adam in deep learning. In *Advances in Neural Information Processing Systems (NeurIPS)*, volume 33.

Zhou, Y., Liang, Y., and Zhang, H. (2018a). Generalization error bounds with probabilistic guarantee for SGD in nonconvex optimization. *arXiv preprint arXiv:1802.06903*.

Zhou, Y., Yang, J., Zhang, H., Liang, Y., and Tarokh, V. (2018b). SGD converges to global minimum in deep learning via star-convex path. In *International Conference on Learning Representations (ICLR)*.

Zou, D., Cao, Y., Li, Y., and Gu, Q. (2021). Understanding the generalization of Adam in learning neural networks with proper regularization. *arXiv preprint arXiv:2108.11371*.

Supplementary Materials

A Additional Experimental Results

A.1 ResNet20 Experiments

In this section, we evaluate the performance of our trained optimizers on larger neural networks ResNet-20 on CIFAR-10 dataset. The training loss and testing accuracy are plotted in Figure 5. We can see that the Entropy regularizer is able to outperform other methods in both training loss and testing accuracy, demonstrating its generalization ability on large unseen models. Further note that although the Hessian regularizer may not be preferred in large neural networks, it does perform better than the Entropy regularizer in small networks as we have shown in Figure 2.

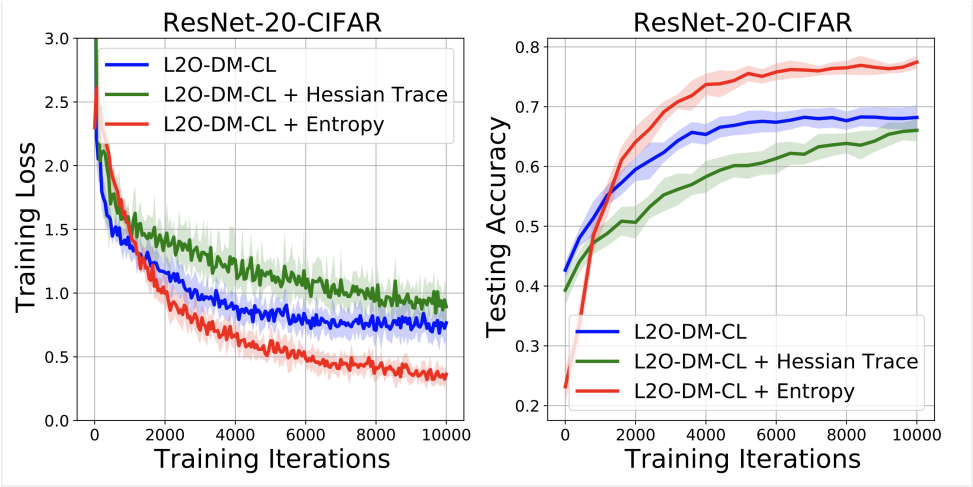


Figure 5: Comparison of the training loss/testing accuracy of ResNet-20 trained using L2O-DM-CL (Chen et al., 2020a) with/without the proposed Hessian/Entropy regularization.

A.2 Wall Clock Comparison between different algorithms

We further conduct an optimizee training time comparison between our methods, analytical optimizers and L2O-DM-CL in Table 1. Note that L2O-DM-CL+Hessian and L2O-DM-CL+Entropy share the same time to train optimizee as L2O-DM-CL. From Table 1, we can see that trained L2O-DM-CL requires only ~ 1.5x time than analytical optimizers in terms of inference time, which is thus time efficient for practical usage.

Table 1: Empirical Time Cost Comparison per Iteration

Methods	SGD	ADAM	L2O (**L2O+Hessian, L2O+Entropy**)
Time (secs)	0.045	0.045	**0.067**

A.3 Accuracy Comparison between different algorithms

We also compare the testing accuracy (%) of our proposed methods with Entropy-SGD (Chaudhari et al., 2017) and SGD with Hessian regularization. The Conv-MNIST results shown in Table 2 are evaluated on L2O-DM-CL and the Conv-CIFAR results shown in Table 3 are evaluated on L2O-Scale. We adopt the same experimental setting as in Section 5 for the Conv-MNIST experiment. We also use same experimental setting for Conv-CIFAR except that the running epochs are set to 100 to investigate whether the performance of trained optimizers would persist in long term.

From these comparisons, we can see that our proposed optimizers (L2O+Hessian, L2O+Entropy) achieve the best performance compared with regularized analytical optimizers. Specifically, in Conv-CIFAR setting as shown in Table 3, our algorithm L2O+Hessian outperforms SGD+Hessian and Entropy-SGD. In Conv-MNIST setting as shown in Table 2, the performances of top three algorithms, i.e. L2O+Entropy, Entropy-SGD and L2O+Hessian, are similar and much better than

Table 2: Additional Testing Accuracy Comparison on Conv-MNIST

Methods	L2O	**L2O+Hessian**	**L2O+Entropy**	SGD	Entropy-SGD	SGD+Hessian
Testing Accuracy	92.74	**97.34**	**97.87**	80.73	97.54	95.37

Table 3: Additional Testing Accuracy Comparison on Conv-CIFAR

Methods	**L2O+Hessian**	Entropy-SGD	SGD	SGD+Hessian
Testing Accuracy	**59.57**	57.73	54.69	51.41

the performances of L2O and SGD+Hessian. Among the top three algorithms, the iteration running time for Entropy-SGD is 0.958 secs while L2O+Hessian and L2O+Entropy only take 0.067 secs as shown in Table 1. Such wall clock comparison shows that L2O+Hessian and L2O+Entropy are more time efficient than Entropy-SGD while achieving the high accuracy, which are preferred for practical usage.

A.4 Accuracy Comparison between different learning rates

We further present the SGD and ADAM results within different learning rates as below:

Table 4: Testing Accuracy Comparison of different SGD learning rates on Conv-MNIST

SGD Learning Rate	0.1	0.01	0.001	0.0001	0.00001
Testing Accuracy	9.81	78.01	80.84	11.09	10.82

Table 5: Testing Accuracy Comparison of different ADAM learning rates on Conv-MNIST

SGD Learning Rate	0.1	0.01	0.001	0.0001	0.00001
Testing Accuracy	9.80	9.95	72.44	55.89	64.84

Based on the Table 4 and table 5, we know that both SGD and Adam are finetuned in terms of learning rates.

B Proof of Theorem 1

B.1 Assumptions

Assumption 1. *Lipschitz properties are assumed on functions $L_{tr}(\theta)$ and $L_{ts}(\theta)$.*

a) $L_{tr}(\theta)$ function is M-Lipschitz, i.e., for any θ_1 and θ_2, $\|L_{tr}(\theta_1) - L_{tr}(\theta_2)\| \leq M\|\theta_1 - \theta_2\|$.

b) $\nabla_\theta L_{tr}(\theta)$ and $\nabla_\theta L_{ts}(\theta)$ are L-Lipschitz, i.e., for any θ_1 and θ_2, $\|\nabla_\theta L_i(\theta_1) - \nabla_\theta L_i(\theta_2)\| \leq L\|\theta_1 - \theta_2\|(i = tr, ts)$.

c) $\nabla^2_\theta L_{tr}(\theta)$ and $\nabla^2_\theta L_{ts}(\theta)$ are ρ-Lipschitz, i.e., for any θ_1 and θ_2, $\|\nabla^2_\theta L_i(\theta_1) - \nabla^2_\theta L_i(\theta_2)\| \leq \rho\|\theta_1 - \theta_2\|(i = tr, ts)$. This assumption also holds for stochastic $\nabla^2_\theta \hat{L}_{tr}(\theta)$ and $\nabla^2_\theta \hat{L}_{ts}(\theta)$.

The above Lipschitz properties also hold for $L(\theta)$, $\nabla_\theta L(\theta)$ and $\nabla^2_\theta L(\theta)$ in Theorem 1.

B.2 Proof of Supporting Lemma

Lemma 1. *Based on Assumption 1 and assuming that function $L(\theta)$ is non-negative and convex, in terms of entropy regularizer $G(\theta; \gamma)$, we have*

$$-G(\theta;\gamma) + mM + \frac{p}{2}\log(2\pi) + \frac{1}{2}\rho m^3 + C(\gamma, p, m) \geq \log(\det(\nabla^2 L(\theta) + \gamma I)) + L(\theta),$$

where m is a constant, $C(\gamma, p, m) = \log \int_{\theta':\|\theta'-\theta\|>m} \exp\left(-\frac{\gamma}{2}\|\theta - \theta'\|^2\right) d\theta'$ and $\theta \in \mathbb{R}^p$.

Proof. We firstly split the integral area $\theta' \in \mathbb{R}^p$ into two parts: $\{\theta' : \|\theta' - \theta\| \leq m\}$ and $\{\theta' : \|\theta' - \theta\| > m\}$. Based on the definition of $G(\theta; \gamma)$, we have

$$
\begin{aligned}
&G(\theta;\gamma)\\
&=\log \int_{\theta'} \exp\Big(-L(\theta') - \frac{\gamma}{2}\|\theta-\theta'\|^2\Big)\,\mathrm{d}\theta'\\
&=\log \int_{\theta':\|\theta'-\theta\|\leq m} \exp\Big(-L(\theta') - \frac{\gamma}{2}\|\theta-\theta'\|^2\Big)\,\mathrm{d}\theta'\\
&\quad+\log \int_{\theta':\|\theta'-\theta\|> m} \exp\Big(-L(\theta') - \frac{\gamma}{2}\|\theta-\theta'\|^2\Big)\,\mathrm{d}\theta'\\
&\overset{(i)}{\leq}\log \int_{\theta':\|\theta'-\theta\|\leq m} \exp\Big(-L(\theta') - \frac{\gamma}{2}\|\theta-\theta'\|^2\Big)\,\mathrm{d}\theta'\\
&\quad+\log \int_{\theta':\|\theta'-\theta\|> m} \exp\Big(-\frac{\gamma}{2}\|\theta-\theta'\|^2\Big)\,\mathrm{d}\theta'\\
&\overset{(ii)}{=}\log \int_{\theta':\|\theta'-\theta\|\leq m} \exp\Big(-L(\theta') - \frac{\gamma}{2}\|\theta-\theta'\|^2\Big)\,\mathrm{d}\theta' + C(\gamma,p,m)\\
&\overset{(iii)}{=}\log \int_{\theta':\|\theta'-\theta\|\leq m} \exp\Big(-L(\theta) - (\theta'-\theta)^T\nabla L(\theta) - \frac{1}{2}(\theta'-\theta)^T\nabla^2 L(\theta'')(\theta'-\theta)\\
&\quad-\frac{\gamma}{2}\|\theta-\theta'\|^2\Big)\mathrm{d}\theta' + C(\gamma,p,m),
\end{aligned}
$$

where (i) follows from the fact that $L(\theta')$ is non-negative, (ii) follows from the fact that $\theta \in \mathbb{R}^p$ and the definiton of $C(\gamma, p, m)$, (iii) follows from Taylor expansion. Note that θ'' satisfies $\|\theta'' - \theta\| \leq \|\theta - \theta'\|$ and $\|\theta'' - \theta'\| \leq \|\theta - \theta'\|$.

Based on Assumption 1, $-(\theta' - \theta)^T \nabla L(\theta) \leq m\|\nabla L(\theta)\| \leq mM$. Then, we obtain

$$
\begin{aligned}
&G(\theta;\gamma)\\
&\leq -L(\theta) + mM + \log \int_{\theta':\|\theta'-\theta\|\leq m} \exp\Big(-\frac{1}{2}(\theta'-\theta)^T\nabla^2 L(\theta'')(\theta'-\theta) - \frac{\gamma}{2}\|\theta-\theta'\|^2\Big)\mathrm{d}\theta' + C(\gamma,p,m)\\
&= -L(\theta) + mM + \log \int_{\theta':\|\theta'-\theta\|\leq m} \exp\Big(-\frac{1}{2}(\theta'-\theta)^T(\nabla^2 L(\theta'')+\gamma I)(\theta'-\theta)\Big)\mathrm{d}\theta' + C(\gamma,p,m)\\
&= -L(\theta) + mM + C(\gamma,p,m)\\
&\quad+\log \int_{\theta':\|\theta'-\theta\|\leq m} \exp\Big(-\frac{1}{2}(\theta'-\theta)^T\Big(\nabla^2 L(\theta'') - \nabla^2 L(\theta) + \nabla^2 L(\theta) + \gamma I\Big)(\theta'-\theta)\Big)\mathrm{d}\theta'\\
&\overset{(i)}{\leq} -L(\theta) + mM + C(\gamma,p,m) + \frac{1}{2}\rho m^3\\
&\quad+\log \int_{\theta':\|\theta'-\theta\|\leq m} \exp\Big(-\frac{1}{2}(\theta'-\theta)^T\left(\nabla^2 L(\theta) + \gamma I\right)(\theta'-\theta)\Big)\mathrm{d}\theta'\\
&\overset{(ii)}{\leq} -L(\theta) + mM + \frac{1}{2}\rho m^3 + \log \int_{\theta'} \exp\Big(-\frac{1}{2}(\theta'-\theta)^T\left(\nabla^2 L(\theta) + \gamma I\right)(\theta'-\theta)\Big)\mathrm{d}\theta' + C(\gamma,p,m),
\end{aligned}
$$

where (i) follows from Assumption 1 and the fact that $\|\theta'' - \theta\| \leq \|\theta - \theta'\|$ and (ii) follows because $\exp(-\frac{1}{2}(\theta' - \theta)^T(\nabla^2 L(\theta) + \gamma I)(\theta' - \theta)) \geq 0$.

Since $L(\theta)$ is convex, we have the fact that $(\nabla^2 L(\theta) + \gamma I)$ is a symmetric and positive-definite matrix. Hence, we obtain

$$
G(\theta;\gamma) \leq -L(\theta) - \log(\det(\nabla^2 L(\theta) + \gamma I)) + C(\gamma,p,m) + mM + \frac{1}{2}\rho m^3 + \frac{p}{2}\log(2\pi).
$$

We rearrange the terms and get

$$
-G(\theta;\gamma) \geq L(\theta) + \log(\det(\nabla^2 L(\theta) + \gamma I)) - C(\gamma,p,m) - mM - \frac{p}{2}\log(2\pi) - \frac{1}{2}\rho m^3.
$$

□

B.3 Proof of Theorem 1

Based on Lemma 1, we have

$$-G(\theta;\gamma) + mM + \frac{p}{2}\log(2\pi) + \frac{1}{2}\rho m^3 + C(\gamma,p,m) \geq \log(\det(\nabla^2 L(\theta) + \gamma I)) + L(\theta).$$

Since $\nabla^2 L(\theta) + \gamma I$ is positive definite and $\lambda_i(\nabla^2 L(\theta) + \gamma I) \geq \gamma$ for any $i = 1, \ldots, p$. Then, based on the definition of Matrix norm $\|\nabla^2 L(\theta) + \gamma I\| = \lambda_{\max}(\nabla^2 L(\theta) + \gamma I)$, we have

$$\|\nabla^2 L(\theta) + \gamma I\|^p \geq \det(\nabla^2 L(\theta) + \gamma I)) \geq \gamma^{p-1}\|\nabla^2 L(\theta) + \gamma I\|.$$

Note that we use $\lambda_i(H)$ to denote the i-th eigenvalue of matrix H. Then,

$$\begin{aligned}\log(\det(\nabla^2 L(\theta) + \gamma I))) \geq& (p-1)\log\gamma + \log\|\nabla^2 L(\theta) + \gamma I\| \\ =& (p-1)\log\gamma + \log(\|\nabla^2 L(\theta)\| + \gamma).\end{aligned}$$

Then, we can obtain

$$\begin{aligned}-G(\theta;\gamma) + mM + \frac{p}{2}\log(2\pi) + \frac{1}{2}\rho m^3 + C(\gamma,p,m) \geq& L(\theta) + (p-1)\log\gamma \\ &+ \log(\|\nabla^2 L(\theta)\| + \gamma).\end{aligned}$$

Hence, we can get a new function $D(x)$ that

$$\|\nabla^2 L(\theta)\| \leq D^{-1}(-G(\theta;\gamma)),$$

where $D(x) = L(\theta) + (p-1)\log\gamma - mM - \frac{p}{2}\log(2\pi) - \frac{1}{2}\rho m^3 - C(\gamma,p,m) + \log(x+\gamma)$. Then, the proof is complete.

C Proof of Theorem 2

C.1 Assumptions

We first define the local basin of θ with the radius d as $D^d(\theta) = \{\theta' : \|\theta - \theta'\|_2 \leq d\}$. As have been observed widely in training a variety of machine learning objectives, the convergent point enters into a local neighborhood where the strong convexity (or similar properties such as gradient dominance condition, reguarity condition, etc) holds (Du et al., 2019; Li and Yuan, 2017; Zhou et al., 2018b; Safran and Shamir, 2016; Milne, 2019). We thus make the following assumption on the geometry of the meta-training function.

Assumption 2. *We assume that there exist a a local basin $D^d(\theta_{tr}^T(\phi))(d > 0)$ of the convergence point $\theta_{tr}^T(\phi)$ that in such local basin, $L_{tr}(\theta)$ and $\hat{L}_{tr}(\theta)$ are μ-strongly convex w.r.t. θ. Futhermore, there exist a unique optimal point θ_{tr}^* of function $L_{tr}(\theta)$ and a optimal point $\hat{\theta}_{tr}^*$ of function $\hat{L}_{tr}(\theta)$ in local basin $D^d(\theta_{tr}^T(\phi^*))$.*

We further adopt the following assumptions introduced in Mei et al. (2018), in order to guarantee the similarity between the landscape of the empirical and population objective functions.

Assumption 3. *Similarly as in Mei et al. (2018), we assume the loss gradient $\nabla l_{tr}(\theta;\xi)$ is τ^2-sub-Gaussian, i.e., for any $\varrho \in \mathbb{R}^p$, and $\theta \in D^r(0)$ where $D^r(0) \equiv \{\theta \in \mathbb{R}^p, \|\theta\|_2 \leq r\}$,*

$$\mathbb{E}\{\exp(\langle \varrho, \nabla l_{tr}(\theta;\xi) - \mathbb{E}[\nabla l_{tr}(\theta;\xi)]\rangle)\} \leq \exp\left(\frac{\tau^2\|\varrho\|^2}{2}\right).$$

Meanwhile, we assume the loss Hessian is τ^2-sub-exponential, i.e., for any $\varrho \in D^1(0)$, and $\theta \in D^r(0)$,

$$\xi_{\varrho,\theta} \equiv \langle \varrho, \nabla^2 l_{tr}(\theta;\xi)\varrho\rangle, \quad \mathbb{E}\Big\{\exp\left(\frac{1}{\tau^2}|\xi_{\varrho,\theta} - \mathbb{E}\xi_{\varrho,\theta}|\right)\Big\} \leq 2,$$

and there exists a constant c_h such that $L \leq \tau^2 p^{c_h}, \rho \leq \tau^3 p^{c_h}$.

Assumption 4. *We assume functions $L_{tr}(\theta)$ is (ϵ,η)-strongly Morse in $D^r(0)$, i.e., if $\|\nabla L_{tr}(\theta)\|_2 > \epsilon$ for $\|\theta\|_2 = r$ and, for any $\theta \in \mathbb{R}^p$, $\|\theta\|_2 < r$, the following holds:*

$$\|\nabla L_{tr}(\theta)\|_2 \leq \epsilon \Rightarrow \min_{i\in[p]} |\lambda_i(\nabla^2 L_{tr}(\theta))| \geq \eta,$$

where $\lambda_i(\nabla^2 L_{tr}(\theta))$ denotes the i-th eigenvalue of $\nabla^2 L_{tr}(\theta)$. We further make the assumption that the local basins $D^d(\theta_i^T(\phi^*))(i = tr, ts)$ of convergence points $\theta_i^T(\phi^*)(i = tr, ts)$ are in $D^r(0)$.

C.2 Proof of Supporting Lemmas

Lemma 2 (Restatement of Theorem 1(b) in Mei et al. (2018))**.** *We assume θ^* corresponding to $\hat{\theta}^*$ in local basin. Based on Assumptions 1 and 3, there exists a universal constant C_0, and we let $C = C_0 \max\{c_h, \log(r\tau/\delta), 1\}$. If $N \geq Cp\log p$, then we have*

$$\sup_{\theta \in D^p(r)} \|\nabla^2 \hat{L}(\theta) - \nabla^2 L(\theta)\| \leq \tau^2 \sqrt{\frac{Cp\log N}{N}},$$

with probability at least $1-\delta$.

Lemma 3 (Restatement of Theorem 2 in Mei et al. (2018))**.** *Based on Assumptions 1, 3 and 4, we set C as in Lemma 2, assume that θ^* is corresponding to $\hat{\theta}^*$, and let $N \geq 4Cp\log N/\eta_*^2$ where $\eta_*^2 = \min\{(\epsilon^2/\tau^2), (\eta^2/\tau^4), (\eta^4/(L^2\tau^2))\}$. Then, for each corresponding $\hat{\theta}^*$ and θ^*, we have*

$$\|\hat{\theta}^* - \theta^*\|_2 \leq \frac{2\tau}{\eta}\sqrt{\frac{Cp\log N}{N}},$$

with probability at least $1-\delta$.

Lemma 4. *Suppose Assumptions 1 and 2 hold. Then, we have*

$$\|\theta_{tr}^T(\phi^*) - \hat{\theta}_{tr}^*\| \leq \sqrt{\frac{L}{\mu}}\left(\frac{L-\mu}{L+\mu}\right)^{T-T'} \|\theta_{tr}^{T'}(GD) - \hat{\theta}_{tr}^*\|, \tag{11}$$

where T' is the minimum that after T' gradient descent updates, the updated optimizee parameter $\theta_{tr}^{T'}(GD)$ locates into the local basin of $\theta_{tr}^T(\phi^*)$ and GD refers to Gradient Descent.

Proof. Since the local basin is μ-strongly convex and $\hat{\theta}_{tr}^*$ is the optimal point of smooth function $\hat{L}_{tr}(\theta)$ in local basin. Then, we have

$$\hat{L}_{tr}(\theta_{tr}^T(\phi^*)) - \hat{L}_{tr}(\hat{\theta}_{tr}^*) \geq \frac{\mu}{2}\|\theta_{tr}^T(\phi^*) - \hat{\theta}_{tr}^*\|^2.$$

Furthermore, we rearrange the terms and obtain

$$\begin{aligned}
\|\theta_{tr}^T(\phi^*) - \hat{\theta}_{tr}^*\| \leq & \sqrt{\frac{2}{\mu}(\hat{L}_{tr}(\theta_{tr}^T(\phi^*)) - \hat{L}_{tr}(\hat{\theta}_{tr}^*))} \\
\overset{(i)}{\leq} & \sqrt{\frac{2}{\mu}(\hat{L}_{tr}(\theta_{tr}^T(GD)) - \hat{L}_{tr}(\hat{\theta}_{tr}^*))} \\
\overset{(ii)}{\leq} & \sqrt{\frac{2}{\mu}\frac{L}{2}\|\theta_{tr}^T(GD) - \hat{\theta}_{tr}^*\|^2} \\
\leq & \sqrt{\frac{L}{\mu}}\|\theta_{tr}^T(GD) - \hat{\theta}_{tr}^*\| \\
\overset{(iii)}{\leq} & \sqrt{\frac{L}{\mu}}\left(\frac{L-\mu}{L+\mu}\right)^{T-T'} \|\theta_{tr}^{T'}(GD) - \hat{\theta}_{tr}^*\|,
\end{aligned}$$

where (i) follows because $\phi^* = \arg\min_\phi \hat{L}_{tr}(\theta_{tr}^T(\phi))$ and $\theta_{tr}^T(GD)$ locates in the local basin of $\hat{\theta}_{tr}^*$, (ii) follows from Assumption 1 which implies $\hat{L}_{tr}(\theta) \leq \hat{L}_{tr}(\hat{\theta}_{tr}^*) + \langle \nabla_\theta \hat{L}_{tr}(\hat{\theta}_{tr}^*), \theta - \hat{\theta}_{tr}^*\rangle + \frac{L}{2}\|\theta - \hat{\theta}_{tr}^*\|^2$ and the fact that $\hat{\theta}_{tr}^* = \arg\min_\theta \hat{L}_{tr}(\theta)$ which implies $\nabla_\theta \hat{L}_{tr}(\hat{\theta}_{tr}^*) = 0$, and (iii) follows if we set step size of GD as $\frac{2}{\mu+L}$. □

Lemma 5. *Based on Assumptions 1, 2, 3 and 4, we let $N \geq \max\{Cp\log p, 4Cp\log N/\eta_*^2\}$ where $C = C_0\max\{c_h, 1, log(\frac{r\tau}{\delta})\}$, $\eta_*^2 = \min\{\frac{\epsilon^2}{\tau^2}, \frac{\eta^2}{\tau^4}, \frac{\eta^4}{\rho^2\tau^2}\}$, C_0 is an universal constant. Then, with probability at least $1-2\delta$ we have*

$$\|\nabla^2_\theta L_{ts}(\theta^*_{ts})\| \leq \rho\left(\frac{2\tau}{\eta}\sqrt{\frac{Cp\log N}{N}} + \sqrt{\frac{L}{\mu}}\left(\frac{L-\mu}{L+\mu}\right)^{T-T'}\|\theta^{T'}_{tr}(GD) - \hat{\theta}^*_{tr}\|\right) + \Delta^*_H + \tau^2\sqrt{\frac{Cp\log N}{N}} + B_{Hessian}(\lambda),$$

where $\Delta^*_H = \|\nabla^2_\theta L_{ts}(\theta^*_{ts}) - \nabla^2_\theta L_{tr}(\theta^*_{tr})\|$, T' *is defined in Lemma 4 and GD refers to Gradient Descent.*

Proof. Firstly, we bound $\|\nabla^2_\theta L_{ts}(\theta^*_{ts})\|$ as following:

$$\begin{aligned}\|\nabla^2_\theta L_{ts}(\theta^*_{ts})\| \leq& \|\nabla^2_\theta L_{ts}(\theta^*_{ts}) - \nabla^2_\theta L_{tr}(\theta^*_{tr})\| + \|\nabla^2_\theta L_{tr}(\theta^*_{tr}) - \nabla^2_\theta \hat{L}_{tr}(\theta^*_{tr})\| \\ &+ \|\nabla^2_\theta \hat{L}_{tr}(\theta^*_{tr}) - \nabla^2_\theta \hat{L}_{tr}(\hat{\theta}^*_{tr})\| + \|\nabla^2_\theta \hat{L}_{tr}(\hat{\theta}^*_{tr}) - \nabla^2_\theta \hat{L}_{tr}(\theta^T_{tr}(\phi^*))\| \\ &+ \|\nabla^2_\theta \hat{L}_{tr}(\theta^T_{tr}(\phi^*))\|,\end{aligned}$$

where θ^*_{tr} is corresponding to $\hat{\theta}^*_{tr}$ in the same local basin of $\theta^T_{tr}(\phi^*)$.

Based on the constrained problem formulation in eq. (8), the optimal optimizer parameter ϕ^* is equivalent to the following:

$$\phi^* = \arg\min_\phi \hat{L}_{tr}(\theta^T_{tr}(\phi)) \text{ subject to } \nabla^2_\theta \hat{L}_{tr}(\theta^T_{tr}(\phi)) \leq B_{\text{Hessian}}(\lambda).$$

Thus, we obtain $\|\nabla^2_\theta \hat{L}_{tr}(\theta^T_{tr}(\phi^*))\| \leq B_{\text{Hessian}}(\lambda)$. Furthermore, if we let $N \geq Cp\log p$ where $C = C_0\max\{c_h, 1, log(\frac{r\tau}{\delta})\}$ and C_0 is an universal constant, based on Lemmas 2, 3 and 4, and Assumptions 1, we have

$$\|\nabla^2_\theta L_{ts}(\theta^*_{ts})\| \leq \rho\left(\|\theta^*_{tr} - \hat{\theta}^*_{tr}\| + \sqrt{\frac{L}{\mu}}\left(\frac{L-\mu}{L+\mu}\right)^{T-T'}\|\theta^{T'}_{tr}(GD) - \hat{\theta}^*_{tr}\|\right) + \Delta^*_H + \tau^2\sqrt{\frac{Cp\log N}{N}} + B_{\text{Hessian}}(\lambda),$$

with probability at least $1-\delta$.

Furthermore, if we assume $N \geq \max\{4Cp\log N/\eta^2_*, Cp\log p\}$ where $\eta^2_* = \min\{\frac{\epsilon^2}{\tau^2}, \frac{\eta^2}{\tau^4}, \frac{\eta^4}{\rho^2\tau^2}\}$, based on Lemma 3, we have

$$\|\nabla^2_\theta L_{ts}(\theta^*_{ts})\| \leq \rho\left(\frac{2\tau}{\eta}\sqrt{\frac{Cp\log N}{N}} + \sqrt{\frac{L}{\mu}}\left(\frac{L-\mu}{L+\mu}\right)^{T-T'}\|\theta^{T'}_{tr}(GD) - \hat{\theta}^*_{tr}\|\right) + B_{\text{Hessian}}(\lambda) + \Delta^*_H + \tau^2\sqrt{\frac{Cp\log N}{N}},$$

with probability at least $1-2\delta$. □

Lemma 6. *Based on Assumptions 2, 1, 3, and 4, we let* $N \geq 4Cp\log N/\eta^2_*$ *where* C *and* η^2_* *are defined in Lemma 3. Then, with probability at least* $1-\delta$*, we have*

$$\|\theta^T_{ts}(\phi^*) - \theta^*_{ts}\| \leq \Delta^*_T + \sqrt{\frac{L}{\mu}}\left(\frac{L-\mu}{L+\mu}\right)^{T-T'}\|\theta^{T'}_{tr}(GD) - \hat{\theta}^*_{tr}\| + \frac{2\tau}{\eta}\sqrt{\frac{Cp\log N}{N}} + \Delta^*_\theta,$$

where $\Delta^*_\theta = \|\theta^*_{tr} - \theta^*_{ts}\|$, $\Delta^*_T = \|\theta^T_{ts}(\phi^*) - \theta^T_{tr}(\phi^*)\|$ *,*T' *is defined in Lemma 4 and GD refers to Gradient Descent.*

Proof. Based on triangle inequality, we obtain

$$\begin{aligned}
&\|\theta_{ts}^T(\phi^*) - \theta_{ts}^*\| \\
&\leq \|\theta_{ts}^T(\phi^*) - \theta_{tr}^T(\phi^*)\| + \|\theta_{tr}^T(\phi^*) - \hat{\theta}_{tr}^*\| + \|\hat{\theta}_{tr}^* - \theta_{tr}^*\| + \|\theta_{tr}^* - \theta_{ts}^*\| \\
&\overset{(i)}{\leq} \Delta_T^* + \|\theta_{tr}^T(\phi^*) - \theta_N^{*(1)}\| + \|\hat{\theta}_{tr}^* - \theta_{tr}^*\| + \|\theta_{tr}^* - \theta_{ts}^*\| \\
&\overset{(ii)}{\leq} \Delta_T^* + \sqrt{\frac{L}{\mu}}\left(\frac{L-\mu}{L+\mu}\right)^{T-T'} \|\theta_{tr}^{T'}(GD) - \hat{\theta}_{tr}^*\| + \|\hat{\theta}_{tr}^* - \theta_{tr}^*\| + \Delta_\theta^*,
\end{aligned}$$

where (i) follows from definition of Δ_T^*, (ii) follows from Lemma 4 and definition of Δ_θ^*. Based on Lemma 3, if we let $N \geq 4Cp\log N/\eta_*^2$. Then, with probability at least $1-\delta$, we have

$$\begin{aligned}
\|\theta_{ts}^T(\phi^*) - \theta_{ts}^*\| \leq& \Delta_T^* + \sqrt{\frac{L}{\mu}}\left(\frac{L-\mu}{L+\mu}\right)^{T-T'} \|\theta_{tr}^{T'}(GD) - \hat{\theta}_{tr}^*\| + \frac{2\tau}{\eta}\sqrt{\frac{Cp\log N}{N}} + \Delta_\theta^* \\
=& \Delta_T^* + \Delta_\theta^* + \mathcal{O}(w^{T-T'}) + \mathcal{O}(\sqrt{\frac{C\log N}{N}}),
\end{aligned}$$

where $w = \frac{L-\mu}{L+\mu}$. □

C.3 Proof of Theorem 2

Generalization loss is defined as below:

$$\begin{aligned}
&L_{ts}(\theta_{ts}^T(\phi^*)) - L_{ts}(\theta_{ts}^*) \\
&\overset{(i)}{=} (\theta_{ts}^T(\phi^*) - \theta_{ts}^*)^T\nabla_\theta L_{ts}(\theta_{ts}^*) + \frac{1}{2}(\theta_{ts}^T(\phi^*) - \theta_{ts}^*)^T\nabla_\theta^2 L_{ts}(\theta')(\theta_{ts}^T(\phi^*) - \theta_{ts}^*) \\
&\overset{(ii)}{=} \frac{1}{2}(\theta_{ts}^T(\phi^*) - \theta_{ts}^*)^T\nabla_\theta^2 L_{ts}(\theta')(\theta_{ts}^T(\phi^*) - \theta_{ts}^*) \\
&= \frac{1}{2}(\theta_{ts}^T(\phi^*) - \theta_{ts}^*)^T(\nabla_\theta^2 L_{ts}(\theta') - \nabla_\theta^2 L_{ts}(\theta_{ts}^*) + \nabla_\theta^2 L_{ts}(\theta_{ts}^*))(\theta_{ts}^T(\phi^*) - \theta_{ts}^*) \\
&\leq \frac{1}{2}\|\theta_{ts}^T(\phi^*) - \theta_{ts}^*\|^2(\|\nabla_\theta^2 L_{ts}(\theta') - \nabla_\theta^2 L_{ts}(\theta_{ts}^*)\| + \|\nabla_\theta^2 L_{ts}(\theta_{ts}^*)\|) \\
&\overset{(iii)}{\leq} \frac{1}{2}\rho\|\theta_{ts}^T(\phi^*) - \theta_{ts}^*\|^3 + \frac{1}{2}\|\nabla_\theta^2 L_{ts}(\theta_{ts}^*)\|\|\theta_{ts}^T(\phi^*) - \theta_{ts}^*\|^2 \\
&\leq \frac{1}{2}\|\theta_{ts}^T(\phi^*) - \theta_{ts}^*\|^2(\rho\|\theta_{ts}^T(\phi^*) - \theta_{ts}^*\| + \|\nabla_\theta^2 L_{ts}(\theta_{ts}^*)\|),
\end{aligned}$$

where (i) follows from Taylor expansion and θ' follows from the conditions that $\|\theta' - \theta_{ts}^T(\phi^*)\| \leq \|\theta_{ts}^* - \theta_{ts}^T(\phi^*)\|$ and $\|\theta' - \theta_{ts}^*\| \leq \|\theta_{ts}^* - \theta_{ts}^T(\phi^*)\|$, (ii) follows because $\nabla_\theta L_{ts}(\theta_{ts}^*) = 0$, and (iii) follows from Assumption 1 and the fact that $\|\theta' - \theta_{ts}^*\| \leq \|\theta_{ts}^* - \theta_{ts}^T(\phi^*)\|$.

Based on Lemmas 5 and 6, if we let $N \geq \max\{4Cp\log N/\eta_*^2, Cp\log p\}$ where C and η_*^2 are defined in Lemma 5. Then, with probability at least $1-2\delta$, we have

$$\begin{aligned}
&\rho\|\theta_{ts}^T(\phi^*) - \theta_{ts}^*\| + \|\nabla_\theta^2 L_{ts}(\theta_{ts}^*)\| \\
&\leq \rho\Big(\Delta_T^* + \sqrt{\frac{L}{\mu}}\left(\frac{L-\mu}{L+\mu}\right)^{T-T'} \|\theta_{tr}^{T'}(GD) - \hat{\theta}_{tr}^*\| + \frac{2\tau}{\eta}\sqrt{\frac{Cp\log N}{N}} + \Delta_\theta^*\Big) \\
&\quad + \tau^2\sqrt{\frac{Cp\log N}{N}} + \rho\left(\frac{2\tau}{\eta}\sqrt{\frac{Cp\log N}{N}} + \sqrt{\frac{L}{\mu}}\left(\frac{L-\mu}{L+\mu}\right)^{T-T'} \|\theta_{tr}^{T'}(GD) - \hat{\theta}_{tr}^*\|\right) + \Delta_H^* + B(\lambda) \\
&= \rho\Delta_T^* + 2\rho\sqrt{\frac{L}{\mu}}\left(\frac{L-\mu}{L+\mu}\right)^{T-T'} \|\theta_{tr}^{T'}(GD) - \hat{\theta}_{tr}^*\| + \left(\frac{4\rho\tau}{\eta} + \tau^2\right)\sqrt{\frac{Cp\log N}{N}} + \rho\Delta_\theta^* + \Delta_H^* + B_{\text{Hessian}}(\lambda) \\
&= B_{\text{Hessian}}(\lambda) + \rho\Delta_T^* + \rho\Delta_\theta^* + \Delta_H^* + \mathcal{O}(w^{T-T'}) + \mathcal{O}(\sqrt{\frac{C\log N}{N}}),
\end{aligned}$$

where $w = \frac{L-\mu}{L+\mu}$, $\Delta_H^* = \|\nabla_\theta^2 L_{ts}(\theta_{ts}^*) - \nabla_\theta^2 L_{tr}(\theta_{tr}^*)\|$, $\Delta_\theta^* = \|\theta_{tr}^* - \theta_{ts}^*\|$, $\Delta_T^* = \|\theta_{ts}^T(\phi^*) - \theta_{tr}^T(\phi^*)\|$.

Then, we have

$$
\begin{aligned}
&L_{ts}(\theta_{ts}^T(\phi^*)) - L_{ts}(\theta_{ts}^*) \\
&\quad \leq \frac{1}{2}\|\theta_{ts}^T(\phi^*) - \theta_{ts}^*\|^2(\rho\|\theta_{ts}^T(\phi^*) - \theta_{ts}^*\| + \|\nabla_\theta^2 L_{ts}(\theta_{ts}^*)\|) \\
&\quad \leq \frac{1}{2}\left(\Delta_T^* + \Delta_\theta^* + \mathcal{O}(w^{T-T'}) + \mathcal{O}(\sqrt{\frac{C\log N}{N}})\right)^2 \left(B_{\text{Hessian}}(\lambda) + \Delta_1^* + \mathcal{O}(w^{T-T'}) + \mathcal{O}(\sqrt{\frac{C\log N}{N}})\right)
\end{aligned}
$$

with probability at least $1 - 2\delta$ where $\Delta_1^* = \rho\Delta_T^* + \rho\Delta_\theta^* + \Delta_H^*$. Then, the proof is complete.