Demystify Hyperparameters for Stochastic Optimization with Transferable Representations

Jianhui Sun
University of Virginia, js9gu@virginia.edu

Mengdi Huai
University of Virginia, mh6ck@virginia.edu

Kishlay Jha
University of Virginia, kj6ww@virginia.edu

Aidong Zhang
University of Virginia, aidong@virginia.edu

ABSTRACT

This paper studies the convergence and generalization of a large class of Stochastic Gradient Descent (SGD) momentum schemes, in both learning from scratch and transferring representations with fine-tuning. Momentum-based acceleration of SGD is the default optimizer for many deep learning models. However, there is a lack of general convergence guarantees for many existing momentum variants in conjunction with *stochastic* gradient. It is also unclear how the momentum methods may affect the *generalization* error. In this paper, we give a unified analysis of several popular optimizers, e.g., Polyak's heavy ball momentum and Nesterov's accelerated gradient. Our contribution is threefold. First, we give a unified convergence guarantee for a large class of momentum variants in the *stochastic* setting. Notably, our results cover both convex and nonconvex objectives. Second, we prove a generalization bound for neural networks trained by momentum variants. We analyze how hyperparameters affect the generalization bound and consequently propose guidelines on how to tune these hyperparameters in various momentum schemes to generalize well. We provide extensive empirical evidence to our proposed guidelines. Third, this study fills the vacancy of a formal analysis of fine-tuning in literature. To our best knowledge, our work is the first systematic generalizability analysis on momentum methods that cover both learning from scratch and fine-tuning. Our codes are available [1].

CCS CONCEPTS

• **Theory of computation** → **Sample complexity and generalization bounds**; • **Mathematics of computing** → *Nonconvex optimization.*

KEYWORDS

Deep Learning Optimization; Fine-tuning; Stochastic Gradient Descent; AutoML; Generalization Bound

[1]https://github.com/jsycsjh/Demystify-Hyperparameters-for-Stochastic-Optimization-with-Transferable-Representations

KDD '22, August 14–18, 2022, Washington, DC, USA

ACM ISBN 978-1-4503-9385-0/22/08.
https://doi.org/10.1145/3534678.3539298

ACM Reference Format:
Jianhui Sun, Mengdi Huai, Kishlay Jha, and Aidong Zhang. 2022. Demystify Hyperparameters for Stochastic Optimization with Transferable Representations. In *Proceedings of the 28th ACM SIGKDD Conference on Knowledge Discovery and Data Mining (KDD '22), August 14–18, 2022, Washington, DC, USA.* ACM, New York, NY, USA, 11 pages. https://doi.org/10.1145/3534678.3539298

1 INTRODUCTION

Gradient based method in conjunction with a *momentum* term is the optimizer of choice in many applications. On the one hand, classical momentum methods (e.g. Polyak's 'heavy ball' momentum [33] and Nesterov's accelerated gradient [31]) are known to achieve faster convergence guarantees than gradient descent for smooth and strongly convex objectives with *deterministic* gradients (gradient computed on the full training dataset) [32, 34]. On the other hand, stochastic gradient descent (SGD) possesses both practical and theoretical appeals over *batch* gradient descent [3], and has now played a central role in training deep learning models. Therefore, momentum in conjunction with *stochastic* gradient has been extensively used.

Since [40] first demonstrated the critical role of well-tuned stochastic momentum method in deep learning optimization, stochastic momentum has been used so pervasively that deep learning models are usually optimized by SGD with momentum instead of vanilla SGD by default. Notably, momentum has not only exhibited convincing performances in accelerating *training* loss minimization, it has also been reported to achieve better generalizability in over-parameterized neural networks than adaptive gradient algorithms [16, 42].

Apart from the classical stochastic variant of heavy ball momentum (SHB) and Nesterov's accelerated gradient (NAG), a growing number of momentum variants in *stochastic* setting have been proposed to train deep learning models to either accommodate objective functions with more general geometry, or to improve training stability and generalization ability in specific settings[2]. Despite empirical successes reported for many momentum variants in a wide spectrum of tasks, how different forms of momentum variants and their associated hyperparameters affect (1) the convergence properties and (2) the generalization abilities, is largely an open question in *stochastic* setting.

More importantly, a growing number of applications rely on transferring representation pretrained from a source task where labeled data is abundant and computational capacity is sufficient,

[2]An incomplete list of momentum schemes will include Synthesized Nesterov Variants (SNV) [19], PID control (PID) [1], Accelerated Stochastic Gradient Method (ASGD) [18], Triple Momentum [41], and Quasi-Hyperbolic Momentum (QHM) [26].

and then fine-tune on target task[3]. However, despite the enormous popularity of transferring representation, there is a lack of rigorous understanding of how well momentum variants converge and generalize in transfer learning with fine-tuning.

To tackle the aforementioned problems, this paper studies a general framework, the Quasi-Hyperbolic Momentum (QHM) method [26] with constant momentum parameters, in the stochastic approximation setting (unbiased gradients with bounded variance), in both training from scratch and fine-tuning paradigms. Note that QHM has a general formulation that could cover a large family of popular momentum methods, e.g., SHB, NAG, ASGD, and SNV. Therefore, our unified analysis includes a large class of momentum schemes as special cases.

Our main contributions could be summarized as follow:

- We derive a unified convergence guarantee for a family of momentum methods, for both convex and nonconvex objectives. Our nonconvex analysis does not require the restrictive 'bounded gradient' assumption used in the existing literature.
- We prove a generalization bound for neural networks trained by momentum variants. To our best knowledge, this is the first systematic generalizability analysis of momentum schemes. Based on the bound, we theoretically justify some training heuristics from existing empirical works [4], and further motivate novel training guidelines to new momentum methods (e.g., PID and SNV). Our empirical experiments test across different data, models, and optimizers, which all verify the effectiveness of our proposed guidelines.
- Our analysis is the first work that provides a rigorous understanding of how momentum methods converge and generalize in learning both with and without fine-tuning.

In Section 2, we formally introduce the definitions of learning from scratch/fine-tuning, momentum schemes, and generalization error, that are pertinent to this work. In Section 3, we provide the convergence guarantee of QHM, for both strongly convex and general nonconvex objective functions. In Section 4, we theoretically connect the generalization error of momentum schemes with hyperparameters, both with and without fine-tuning. In Section 5, we justify existing tuning heuristics and propose new tuning rules. We conduct extensive experiments to verify our proposed guidelines in Section 6. Related works are discussed in Section 7. Proofs of all our theorems and corollaries are presented in Appendix (Section 9).

2 BACKGROUND

2.1 Learning from Scratch vs. Fine-tuning

Let X be the input space and C be the label space. Suppose $h_T : X \mapsto C$ is the ground truth for the target task T. The learning goal is to find a hypothesis h such that it minimizes the target risk $\mathcal{R}_T(h) \triangleq \mathbb{E}_{x\sim P_T(X)}[l(h_T(x), h(x))]$, where $P_T(X)$ is the input data distribution on target task, and l is a loss function.

[3]Examples of powerful pretrained models include Caffe Model Zoo [14] and BERT [4].
[4]e.g., regularizing the distance between fine-tuned weights and the pre-trained weights, linearly scaling batch size with learning rate, NAG generalizing better than SHB in many experimental settings, and monotonically increasing momentum parameter $\beta \to 1$ [15, 36]

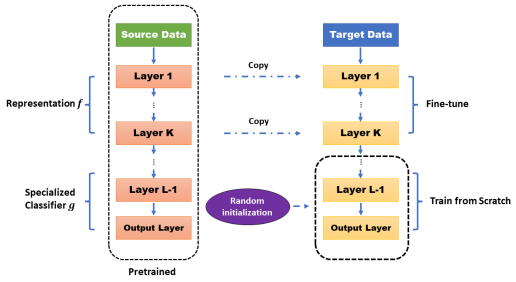

Figure 1: The workflow of fine-tuning. In most cases, a specialized classifier only containing the last output layer is trained from scratch, while representation containing all previous layers is transferred from the source and fine-tuned.

Deep learning models generally first learn the disentangled features of input. Suppose Z is the feature space, $F = \{f : X \mapsto Z\}$ is a class of *representations*, and $G = \{g : Z \mapsto C\}$ is a class of *specialized classifiers*. Any hypothesis is a composite of a representation and specialized classifier, i.e. $h = g \circ f : X \mapsto C$, and the hypothesis class is $H \triangleq \{h : \exists f \in F, g \in G, \text{such that}, h = g \circ f\}$. Each hypothesis is parameterized by $\theta = (\theta_f, \theta_g)$. Figure 1 describes the workflow of fine-tuning.

Suppose (x, y) is a labeled data instance, and N is the number of labeled data points in the target task. The traditional Frequentist learning paradigm views model parameter θ as fixed but unknown values and do not attach any probabilities to these learnable parameters. In contrast, the Bayesian perspective shifts to studying a distribution of every possible setting of parameters instead of betting on one single setting of parameters to manage model uncertainty, and has proven to be increasingly powerful in many settings. In the Bayesian framework, θ is assumed to follow some prior distribution P (reflects our prior knowledge of model parameters), and through the iterative optimization, the distribution of θ shifts to $\{Q_k\}_{k\geq 0}$, and converges to posterior distribution Q (reflects our knowledge of model parameters after learning with N data points). The mode of the posterior distribution is typically regarded as the final model parameter.

The generalization bound could therefore be defined as follows:

$$\mathcal{E} \triangleq |\mathcal{R}(Q) - \hat{\mathcal{R}}(Q)|. \tag{1}$$

where $\mathcal{R}(Q) \triangleq \mathbb{E}_{\theta\sim Q}\mathbb{E}_{x\sim P_T(X)} l(h_\theta(x), y)$, and $\hat{\mathcal{R}}(Q) \triangleq \mathbb{E}_{\theta\sim Q} \frac{1}{N} \sum_{j=1}^{N} l(h_\theta(x_j), y_j)$.

In the Bayesian framework, the stochastic hypothesis $\tilde{h}_{g\circ f}$ is a distribution over hypothesis class H and is associated with the Frequentist hypothesis $h = g \circ f$ (i.e., the mode of $\tilde{h}_{g\circ f}$ is $g \circ f$); the stochastic hypothesis class is $\tilde{H}_{G\circ F} \triangleq \{\tilde{h}_{g\circ f} : \exists f \in F, g \in G\}$. Let us introduce the two learning paradigms that we consider in this paper.

Learning from Scratch When N is sufficiently large, we do not need to leverage any external information in learning and could directly train from scratch. We would start from some random initial stochastic hypothesis $\tilde{h}_{g_0\circ f_0}$ (i.e., a distribution centered around random initial hypothesis $g_0 \circ f_0$), and pose no restriction over the

search space $\tilde{H}_{G\circ F} \triangleq \{\tilde{h}_{g\circ f} : \exists f \in F, g \in G$. We would search in $\tilde{H}_{G\circ F}$ with any optimizer minimizing the target risk R_T.

Fine-tuning When target data amount N is not sufficiently large, while there is a source task where labeled data is abundant and we believe these tasks share similar intermediate representation, we could transfer representations learned from a source task to the target task. Formally, we have source task S and the number of labeled data N_S is typically much larger than N. Source risk is defined similarly as $\mathcal{R}_S(h) \triangleq \mathbb{E}_{x\sim P_S(X)}[h_S(x) \neq h(x)]$, where h_S and P_S are the ground truth and input data distribution in source task, respectively. We obtain a minimizer of R_S as $\hat{g}_S \circ \hat{f}_S$ with some optimizer. As we believe target task is similar in internal representation as source task, $\tilde{h}_{\hat{g}_S\circ\hat{f}_S}$ is thus a much better starting point than random initialization $\tilde{h}_{g_0\circ f_0}$. Furthermore, representation in source task is believed to approximate $\hat{f}_S$, and therefore, we will search in $\tilde{H}_{G\circ\hat{F}} \triangleq \{\tilde{h}_{g\circ f} : \exists f \in \hat{F}, g \in G$. $\hat{F} \subseteq F$ and is a region around $\hat{f}_S$ (such restriction is enforced through a penalty term of distance between f and $\hat{f}_S$). As g is the *specialized* classifier specific to the task, and typically takes only a small proportion of all model parameters, no restriction will be posed on function class G.

Both source task and target task are optimized with an optimizer, and in this paper, we focus on Stochastic Gradient Descent (SGD) momentum and its variants.

2.2 Gradient Descent Algorithm and Its Variants

This paper studies a large class of first-order gradient descent algorithms. Recall θ is the parameter to be optimized in the hypothesis, and $\mathcal{R}$ is the risk function. Let $z_i = (x_i, y_i)$ represents a single data instance. $\mathcal{R}_\zeta$ is the empirical risk evaluated by a mini-batch of random samples, $\mathcal{R}_\zeta \triangleq \frac{1}{n_b}\sum_{j\in\zeta}\mathcal{R}_j$, where $\mathcal{R}_j \triangleq l_\theta(z_j)$ is the contribution to risk from j-th data point. ζ represents a mini-batch of random samples and $n_b \triangleq |\zeta|$ represents the batch size. $\hat{g}(\theta) \triangleq \nabla_\theta\mathcal{R}_\zeta$ is the empirical gradient of risk with respect to θ.

We start from the formula of Stochastic Gradient Descent (SGD):

$$\theta_{k+1} = \theta_k - \alpha_k d_k \tag{2}$$

where α_k and d_k are the learning rate and search direction at k-th step, respectively. (mini-batch) SGD [5] uses $\hat{g}_k \triangleq \hat{g}(\theta_k)$ as d_k.

We focus on Quasi-Hyperbolic Momentum methods [26], which could be formulated as:

$$\begin{aligned} d_{k+1} &= (1-\beta_k)\hat{g}_k + \beta_k d_k \\ \theta_{k+1} &= \theta_k - \alpha_k[(1-\nu_k)\hat{g}_k + \nu_k d_k] \end{aligned} \tag{3}$$

More specifically, in this paper, we study QHM with constant parameters, i.e., $\alpha_k = \alpha, \beta_k = \beta, \nu_k = \nu$.

Note that the formulation of Quasi-Hyperbolic Momentum method is very general, and could recover many momentum methods with different specifications of (α, β, ν). For example, QHM recovers plain SGD when $\nu = 0$.

If $\nu = 1$, QHM recovers SHB:

$$d_{k+1} = (1-\beta)\hat{g}_k + \beta d_k \qquad \theta_{k+1} = \theta_k - \alpha d_k \tag{4}$$

where variable d is commonly referred to as the 'momentum buffer'. The exponential discount factor β controls how slowly the momentum buffer is updated.

If $\nu = \beta$, QHM recovers NAG:

$$\begin{aligned} d_{k+1} &= (1-\beta)\hat{g}_k + \beta d_k \\ \theta_{k+1} &= \theta_k - \alpha[(1-\beta)\hat{g}_k + \beta d_k] \end{aligned} \tag{5}$$

From the connection between QHM and SHB (NAG), QHM could be interpreted as a ν-weighted average of the momentum update step and the plain SGD update step. We refer ν as the immediate discount factor.

[26] showed that QHM could recover many other popular momentum schemes, e.g., PID control (PID), Synthesized Nesterov Variants (SNV), Accelerated Stochastic Gradient Method (ASGD), and Triple Momentum, with different α, β, ν specifications. Therefore, our analysis based on QHM could cover a family of momentum methods as special cases.

3 CONVERGENCE OF MOMENTUM SCHEMES IN STOCHASTIC APPROXIMATION SETTING

In this section, we will discuss the dynamics and convergence of the iterates produced by momentum schemes, for both strongly convex and nonconvex objectives. All proofs are deferred to Appendix (Section 9) unless specified otherwise.

3.1 Strongly Convex Objective

Assumption 1 (Locally Quadratic Risk Function [6]). *Suppose the risk function is approximately convex and 2-order differentiable, in the region close to minimum, i.e., there exists a $\delta_0 > 0$, such that $\mathcal{R}(\theta) = \frac{1}{2}(\theta-\theta^*)^T A(\theta-\theta^*)$ if $\|\theta-\theta^*\| \leq \delta_0$, where θ^* is a minimizer of $\mathcal{R}(\theta)$. Here A is the Hessian matrix $\nabla^2_\theta\mathcal{R}$ around minimizer and is positive definite. Without loss of generality, we assume a minimizer of the risk is zero, i.e., $\theta^* = 0$.*

Let us denote the suboptimality of the current iterate as $r_k \triangleq \theta_k - \theta^*$. Following from Assumption 1, the risk function is 2-order differentiable. We denote the Hessian as $\nabla^2\mathcal{R}(\theta)$ in a local region around minimizer θ^*, and the Hessian is positive definite. Let $A_k \triangleq \int_0^1 \nabla^2\mathcal{R}(\theta^* + tr_k)dt$. Denote the gradient noise at iteration k as $\xi_k \triangleq \Delta g(\theta_k)$, where $\Delta g(\theta_k) = \hat{g}(\theta_k) - g(\theta_k)$. Suppose at each step k, gradient noise is Gaussian (the stochastic gradient is a sum of n_b independent, uniformly sampled contributions. Invoking the central limit theorem, we assume that the gradient noise is Gaussian) with mean 0 and covariance $\frac{1}{n_b}\Sigma$. We know $\Delta g(\theta_k)$ is a random variable with mean 0 and bounded variance σ^2, i.e., $\mathbb{E}[\xi_k] = 0$ and $\mathbb{E}\|\xi_k\|^2 \leq \sigma^2$. Here σ^2 could be $\|\Sigma\|$. By the fundamental theorem of calculus, we know:

$$\hat{g}(\theta_k) = A_k r_k + \xi_k \tag{6}$$

[5] As mini-batch GD contains 'one-instance' SGD as a special case, we use 'SGD' to refer to mini-batch GD throughout this paper unless otherwise specified.

[6] Though here we assume quadratic form of risk function, all our results apply to locally smooth and strongly convex objectives. Note that the assumption on locally quadratic structure of loss function, even for extremely nonconvex objectives, could be justified empirically. [21] visualized the loss surfaces for deep structures like ResNet [9] and DenseNet [12], observing quadratic geometry around local minimum in both cases. And certain network architecture designs (e.g., skip connections) could further make neural loss geometry show no noticeable nonconvexity.

Recall the QHM formulation in Equation (3) and apply Equation (6), we could write the recursive pattern of QHM as follows:

$$\begin{bmatrix} d_k \\ r_{k+1} \end{bmatrix} = T_k \begin{bmatrix} d_{k-1} \\ r_k \end{bmatrix} + S\xi_k \tag{7}$$

where T_k and S are functions of (α, β, ν) and A_k:

$$T_k = \begin{bmatrix} \beta I & (1-\beta)A_k \\ -\alpha\nu\beta I & I - \alpha(1-\nu\beta)A_k \end{bmatrix}, S = \begin{bmatrix} (1-\beta)I \\ -\alpha(1-\nu\beta)I \end{bmatrix}$$

With the assumption of locally quadratic risk function, we know $A_k = A$ for all k.

Unrolling the recursion in (7), we have the following:

$$\begin{bmatrix} d_{k-1} \\ r_k \end{bmatrix} = T^k \begin{bmatrix} d_{-1} \\ r_0 \end{bmatrix} + \sum_{j=1}^{k} T^{k-j} S \xi^j \tag{8}$$

where d_{-1} is 0 by default. With Equation (8), we present the convergence result for quadratic functions:

Theorem 1. *Denote the spectral radius of T as $\rho(T)$. Let the smallest and largest eigenvalues of Hessian matrix as: $\mu = \min_i \lambda_i(A)$ and $L = \max_i \lambda_i(A)$. Let $\Delta_\lambda \triangleq (1+\beta-\alpha\lambda+\alpha\nu\beta\lambda)^2 - 4\beta(1-\alpha\lambda+\alpha\lambda\nu)$. Let $\{\theta_k\}_{k\in\mathbb{N}}$ as the sequence from QHM. There exists a vanishing sequence of $\{\epsilon_k\}$, i.e., $\lim_{k\to\infty}\epsilon_k = 0$, such that for all k, the expected optimality gap satisfies,*

$$\mathbb{E}\|\theta_k - \theta^*\|^2 \le \varepsilon_d + \varepsilon_s$$
$$\varepsilon_d = (\rho(T)+\epsilon_k)^{2k}\|\theta_0 - \theta^*\|^2$$
$$\varepsilon_s = \frac{1-\beta^2+\alpha^2(1-\nu\beta)^2}{1-(\rho(T)+\epsilon_k)^2}\sigma^2$$

where $\rho(T) = \max\{\rho_\mu, \rho_L\}$, with:

$$\rho_\lambda = \begin{cases} \frac{1}{2}|1+\beta-\alpha\lambda+\alpha\nu\beta\lambda| + \frac{1}{2}\sqrt{\Delta_\lambda}, & \text{if } \Delta_\lambda \ge 0 \\ \sqrt{\beta(1-\alpha\lambda+\alpha\lambda\nu)}, & \text{otherwise} \end{cases}$$

ε_d and ε_s reflect two sources of expected optimality gap, ε_d is the deterministic approximation error, representing the rate of convergence to local region around minimizer, while ε_s is the stochastic error, describing the irreducible fluctuation around the minimizer due to gradient noise. Both ε_d and ε_s are important in determining the dynamics of QHM. Based on ε_d, the condition to converge to local region of minimizer is $\rho(T) < 1$ as $\lim_{k\to\infty}(\rho(T)+\epsilon_k) = \rho(T)$. Based on ε_s, the fluctuation is a constant factor multiplying noise σ^2, where the constant factor depends on (α, β, ν). Our result extends the deterministic analysis in [6] to stochastic setting. In the deterministic analysis, the second term $\sum_{j=1}^{k} T^{k-j} S\xi_j$ in (8) is ignored.

Remark 3.1 (Connection to Convergence Result in NAG). *Set $\beta = \nu$, and re-scale $\alpha \to \frac{\alpha}{(1-\beta)}$, Theorem 1 will recover the convergence guarantee for stochastic NAG (see Theorem 1 in [2]) [7].*

Remark 3.2 (Estimation of Variance factor). *In Theorem 1, ε_d is easier to handle as we know it is dominated by $\rho(T)$. Let us denote $C(\epsilon) \triangleq \frac{1-\beta^2+\alpha^2(1-\nu\beta)^2}{1-(\rho(T)+\epsilon_k)^2}$, i.e., $\varepsilon_s = C(\epsilon)\sigma^2$. We conduct simulation studies and show how $C(\epsilon)$ changes w.r.t. different model specifications. Specifically, we optimize quadratic functions with QHM. The dimension of parameters is set to be $d = 20$. The Gaussian noise we add is $c \cdot I_d$, and c is fixed at 0.03. Q is the ratio of the largest eigenvalue to the smallest eigenvalue for this quadratic function, i.e., $Q \triangleq \frac{L}{\mu}$. Q is a measure of 'sharpness' at minimum. Figure 2 shows the result of our simulation studies. As we could observe from Figure 2, variance constant factor is smaller when $\beta \to 1$ in both cases. This partially verifies our training heuristic: move β to 1 under the condition of convergence. And we could see that if Q is smaller, the convergence region is larger in area. As smaller Q represents flatter geometry around minimum, it indicates QHM is easier to converge with flatter local geometry.*

[7] The exact formulation of NAG is different in [2], as the learning rate in our QHM formulation re-scales their learning rate: $\alpha \to \frac{\alpha}{(1-\beta)}$.

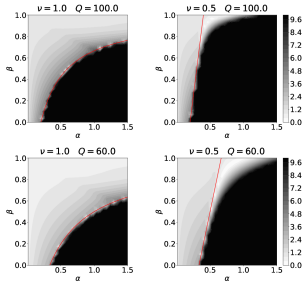

Figure 2: $C(\epsilon)$ in simulation studies. Upper row represents quadratic function with $Q = 100.0$, and lower row represents quadratic function with $Q = 60.0$. We test QHM with $\nu = 1.0, 0.5$. Each pixel represents one single run of QHM with specific α and β. Red line denotes the condition of convergence in Theorem 1. From this figure, we could see QHM is easier to converge with flatter geometry and $\beta \to 1$ will improve the variance constant factor.

3.2 Nonconvex Objective

Assumption 1 (i.e., locally quadratic assumption) in Section 3.1 may not hold for highly over-parameterized neural networks (see e.g. [23]: the set of local minima is often a low-dimensional manifold, instead of isolated points, in the parameter space; and as long as the manifold is not linear, there should be no locally convex region). Therefore, we provide the following convergence guarantee for QHM when optimizing nonconvex functions.

Theorem 2. *Suppose $\mathcal{R}(\theta)$ is L-smooth and not necessarily strongly convex. Denote the expected gradient square as $\{\mathcal{G}_k \triangleq \mathbb{E}[\|g_k\|^2]\}_{k\in\mathbb{N}}$ and $\bar{\mathcal{G}} \triangleq \frac{1}{k}\Sigma_{i=1}^{k}\mathcal{G}_k$. We have: $\bar{\mathcal{G}} \le \varepsilon_d + \varepsilon_s$, where,*

$$\varepsilon_d = \frac{2\Big(\mathcal{R}(\theta_1) - \mathcal{R}^*\Big)}{k\alpha}$$
$$\varepsilon_s = \Big(1 + \frac{\beta^2\nu^2}{1+\beta} + \frac{1-\beta}{1+\beta}\beta^2\nu^6 + (1-\beta\nu)^2\nu^4\Big)L\alpha\sigma^2 \tag{9}$$

if $\alpha \le \min\{\frac{1-\beta}{(3+2\beta^2\nu^2+2\nu^4)L}, \frac{\nu(1-\beta)\sqrt{1-\beta}}{2\sqrt{2}\sqrt{\beta+\beta^2}L}\}$.

Remark 3.3 (Rolel of (β, ν)). *In Theorem 2, we do not require the 'bounded gradient' assumption $\|g\| \le G$ for some $G > 0$ [44]. Specifications of (β, ν) mainly affect the radius of stationary distribution ε_s.*

Please refer to Appendix (Section 9) for proof.

4 GENERALIZATION OF MOMENTUM WITH AND WITHOUT FINE-TUNING

In this section, we introduce our main theorem to explain the role of momentum in closing the generalization gap in both learning from scratch and fine-tuning. We will also discuss how the theorem guides practical model training with SGD momentum variants. All proofs are deferred to Appendix (Section 9).

We first introduce the following lemma that describes the limiting behavior of SGD momentum variants.

Lemma 1. *θ optimized by QHM will converge to a stationary distribution $\tilde{h} = \exp\{-\frac{1}{2}\theta^T\Sigma_\theta\theta\}$. Furthermore, the trace $tr(\Sigma_\theta)$ and determinant $\det(\Sigma_\theta)$ of Σ_θ fulfill the following equalities, $tr(\Sigma_\theta) = \frac{\alpha}{2b}tr(\Sigma A^{-1}) + \frac{\alpha^2}{2b}\left(\frac{\nu\beta}{1-\beta}(1-\frac{2(1+\beta-\nu\beta)}{1+\beta}+\frac{1}{2})\right)tr(\Sigma)+O(\alpha^3)$, and $\det(\Sigma_\theta) = (\frac{\alpha}{2b})^d\det(\Sigma A^{-1}) + O(\alpha^2)$.*

With this lemma, we are prepared to give our main theorem.

Theorem 3. *Assume the target task has n_T training data instances. Assume the risk function is locally quadratic, and gradient noise is Gaussian. Suppose the prior distribution of parameters is $\tilde{h}_0$ and posterior distribution is $\tilde{h}$, both of which are some distributions on hypothesis class parameterized by θ. For any positive real $\epsilon \in (0,1)$, the following inequality holds with probability at least $1-\epsilon$,*

$$\mathcal{R}(Q) \le \hat{\mathcal{R}}(Q) + \sqrt{\frac{KL(\tilde{h}||\tilde{h}_0) + \log\frac{1}{\epsilon} + \log n_T + 2}{2n_T - 1}}$$

***Learning from Scratch**: $\tilde{h}_0 = \mathcal{N}(\theta_0, \lambda_0 I_d)$, where $\theta_0 = (\theta_{0,f}, \theta_{0,g})$ is some random initialization for $h = g\circ f$. λ_0 represents the uncertainty in the Gaussian initialization, where d represents the dimension of parameters.*

***Fine-tuning**: $\tilde{h}_0 = \tilde{h}_{\hat{g}_S\circ\hat{f}_S} = \mathcal{N}((\theta_{S,f}, \theta_{S,g}), \lambda_0 I_d)$.*

Invoking Lemma 1, we will have $KL(\tilde{h}||\tilde{h}_0) = \mathcal{E}_1 + \mathcal{E}_2$, where, $\mathcal{E}_1 = \frac{1}{2\lambda_0}\|\theta_0\|_2^2 + \frac{1}{2}d\log\lambda_0 - \frac{1}{2}d$, and $\mathcal{E}_2 = -\frac{1}{2}\log\det(\Sigma A^{-1}) - \frac{1}{2}d\log\frac{\alpha}{2n_b} + \frac{\alpha tr(\Sigma A^{-1})}{4\lambda_0 n_b} + \frac{\alpha^2 tr(\Sigma)}{4\lambda_0 n_b}\frac{(4\nu^2-2\nu-1)\beta^2-2\nu\beta+1}{2(1-\beta^2)}$.

Proof. Please refer to Appendix (Section 9) for proof. □

Let us denote:

$$K \triangleq \frac{\alpha^2\mathrm{tr}(\Sigma)}{4\lambda_0 n_b}\frac{(4\nu^2-2\nu-1)\beta^2-2\nu\beta+1}{2(1-\beta^2)}$$

In the rest of the paper, when we study how (β,ν) affects the generalization bound, we would focus on how K changes with different settings of (β,ν), as it is the only term in Theorem 3 containing (β,ν).

Remark 4.1 (Connection to Generalization Error in SGD). *Generalization error with respect to momentum is less understood, but there are some existing results on vanilla SGD. To recover the generalization error in SGD, we just need to set $\nu = 0$ in Theorem 3. We immediately have the following corollary:*

Corollary 4.1. *Replace $\mathcal{E}_2$ in Theorem 3 with the following formulation will give us the PAC-Bayesian generalization bound for vanilla SGD:*

$$\mathcal{E}_2 = -\frac{1}{2}d\log\frac{\alpha}{2n_b} - \frac{1}{2}\log\det(\Sigma A^{-1}) + \frac{\alpha tr(\Sigma A^{-1})}{4\lambda_0 n_b} + \frac{\alpha^2 tr(\Sigma)}{8\lambda_0 n_b}$$

Note that [8] proved a similar PAC-Bayesian bound for vanilla SGD. However, our bound in this corollary has a nontrivial difference than theirs. Specifically, their bound only includes a first-order term of learning rate, while ours includes one extra second-order term $\frac{\alpha^2 tr(\Sigma)}{8\lambda_0 n_b}$. The difference is important, in that it incurs the following corollary that could not be derived from their first order bound:

Corollary 4.2. *We give the following two statements about the relationship between learning rate and generalization when training deep learning models with sufficiently large d (see Corollary 5.1 for exact size of d):*

- *Increasing learning rate (under the condition of convergence) will give us a better (smaller) generalization bound.*
- *But if we hold the ratio of batch size to learning rate constant, the generalization bound will get worse (larger) if we increase the learning rate.*

This second statement is obvious from our second order bound. However, with their first order bound, the generalization bound remains constant if fixing the ratio $\frac{n_b}{\alpha}$. Our experimental results (see e.g., Table 2) verifies our theoretical findings.

The practical implication is important as a popular training heuristic is to scale batch size $n_b \propto \alpha$ [36]. Controlling the ratio is a good strategy, but our result warns that we still need to restrict the learning rate.

5 GUIDELINES TO IMPROVE GENERALIZATION

With the above Theorem 3, we are ready to provide several useful training heuristics to improve generalization performance.

5.1 Enforce $\theta_{f,T}$ to be close to $\theta_{f,S}$

As $\mathrm{KL}(\tilde{h}||\tilde{h}_0)$ is the main factor that controls the generalization error, the typical L_2 regularization $\|\theta\|_2^2$ is insufficient for fine-tuning. $\mathrm{KL}(\tilde{h}||\tilde{h}_0)$ is determined by $\theta_S - \theta_T$, where θ_T and θ_S are parameters in target task and source task, respectively. Each set of θ is a composition of θ_f and θ_g, where θ_f encodes the representation we expect the tasks to share, and θ_g is task-specific parameters. Add the following regularization is guaranteed to improve the generalization performance according to Theorem 3,

$$\varphi_1\|\theta_{f,S} - \theta_{f,T}\|_2^2 + \varphi_2\|\theta_{g,T}\|_2^2 \tag{10}$$

where φ_1 and φ_2 are two hyperparameters. We penalize the representation in target task far from the source representation while restricting the complexity of specialized classifier. [20, 22, 30] also demonstrated the improvement from regularizing distances between representations.

5.2 Small ratio $\frac{n_b}{\alpha}$ helps *deep* models generalize

A direct application of Theorem 3 is the following corollary:

Corollary 5.1. *If d is larger than $\frac{\alpha tr(\Sigma A^{-1})}{2\lambda_0 n_b} + 2K$, then the correlation between generalization error and $\frac{n_b}{\alpha}$ is positive.*

Proof. Denote $r = \frac{n_b}{\alpha}$ for ease of notation. Let us calculate the derivative of $\mathcal{E}_2$ to r: $\frac{\partial\mathcal{E}_2}{\partial r} = \frac{d}{2r} - \left(\frac{\mathrm{tr}(\Sigma A^{-1})}{4\lambda_0} +$

$\frac{\alpha \mathrm{tr}(\Sigma)}{4\lambda_0} \frac{(4\nu^2-2\nu-1)\beta^2-2\nu\beta+1}{2(1-\beta^2)}\Big) \frac{1}{r^2}$. It is straightforward to verify if we have $d \geq \frac{\alpha \mathrm{tr}(\Sigma A^{-1})}{2\lambda_0 n_b} + 2K$, we could get $\frac{\partial \mathcal{E}_2}{\partial r} \geq 0$, which completes our proof.

□

The condition for d (the number of parameters) is easily satisfied for many overparameterized deep models. It may sound counter-intuitive that large batch size may hurt models to generalize. But it is consistent with our theoretical and empirical evidence if training an overparameterized model. Note that controlling $\frac{n_b}{\alpha}$ has been used in [8, 36] for vanilla SGD, and here we extend it to momentum-based methods.

Corollary 5.1 also tells us if training *shallow* models, the previous relationship does not necessarily hold. Moreover, recalling Corollary 4.2, we could conclude that learning rate should be restricted if we scale n_b with α.

5.3 NAG generalizes better than SHB, when $\beta \to 1$

To recover the generalization error in SHB and NAG, we just need to set $\nu = 1$ and $\nu = \beta$, respectively.

$$K_{\mathrm{SHB}} = \frac{\alpha^2 \mathrm{tr}(\Sigma)}{4\lambda_0 n_b} \frac{1-\beta}{2(1+\beta)} \tag{11}$$

$$K_{\mathrm{NAG}} = \frac{\alpha^2 \mathrm{tr}(\Sigma)}{4\lambda_0 n_b} \frac{1+\beta-2\beta^2-4\beta^3}{2(1+\beta)} \tag{12}$$

We could conclude from (11) and (12): K_{SHB} will be constantly larger than K_{NAG} when $\beta \to 1$. K_{SHB} is smaller than K_{NAG} when $\beta \in (0, \frac{1}{2})$.

Therefore, $\mathcal{E}_{\mathrm{NAG}} < \mathcal{E}_{\mathrm{SHB}}$, in a typical β setting. In other words, NAG provably generalizes better than SHB. [40] compared empirically SHB and NAG in training deep learning models, and concludes that NAG performs better than SHB in most of their experimental settings. [44] studied qualitatively, hypothesizing NAG is better in achieving a tradeoff between speed of convergence and algorithm stability compared to SHB, without giving an explicit theoretical justification. Our analysis here provides an explanation why and when NAG is better.

5.4 Set momentum parameter β close to 1 and ν as 0.5

[26] proposed to adopt a rule of thumb specification of ($\beta_{\mathrm{rot}} = 0.999, \nu_{\mathrm{rot}} = 0.7$). However, from the expression of K, we could easily conclude: $K(\beta \to 1, \nu = 0.5)$ could be arbitrarily small, and thus much smaller than $K(\beta_{\mathrm{rot}} = 0.999, \nu_{\mathrm{rot}} = 0.7)$.

Therefore, we suggest that adopting a large β and setting ν as 0.5, especially when the learning rate is large (e.g., initial stage of training), will potentially improve the generalization ability. Our experiments (see Table 2) show that test accuracy with $\beta_{\mathrm{opt}} = 0.999, \nu_{\mathrm{opt}} = 0.5$ will generally improve over ($\beta_{\mathrm{rot}} = 0.999, \nu_{\mathrm{rot}} = 0.7$), by a non-trivial margin [8].

[8] We do not indicate $\beta_{\mathrm{opt}} = 0.999$, $\nu_{\mathrm{opt}} = 0.5$ is the optimal setting. Our goal is to show training guidance motivated by our bound could in fact improves generalization.

5.5 Guidelines to other momentum schemes

Apart from classical SHB and NAG, Theorem 3 could also guide the training of some recently proposed momentum schemes. Here we study the case of SNV and PID. We briefly introduce the definitions of SNV and PID for ease of reference (see e.g., [1, 19] for more details).

Definition 1 (Synthesized Nesterov Variant (SNV)). *Synthesized Nesterov Variant, parameterized by $\gamma, \beta_1, \beta_2 \in \mathbb{R}$, uses the following update rule:*

$$\begin{aligned} \xi_{k+1} &\leftarrow \xi_k - \gamma \cdot \hat{g}_k + \beta_1(\xi_t - \xi_{t-1}) \\ \theta_{k+1} &\leftarrow \xi_{k+1} + \beta_2(\xi_{k+1} - \xi_k) \end{aligned} \tag{13}$$

Definition 2 (PID Control). *A PID control optimizer, parameterized by $k_P, k_I, k_D \in \mathbb{R}$, uses the update rule:*

$$\begin{aligned} v_k &\leftarrow \beta \cdot v_{k-1} + (1-\beta)(e_k - e_{k-1}) \\ e_k &\leftarrow -\hat{g}_k \quad w_k \leftarrow w_{k-1} + e_k \\ \theta_{k+1} &\leftarrow \theta_0 + k_P \cdot e_t + k_I \cdot w_t + k_D \cdot v_t \end{aligned} \tag{14}$$

Acknowledging the connection to QHM, we make two practical training guidelines to help improve the generalization ability of SNV and PID.

Corollary 5.2. *Under the condition of convergence, the following two guidelines could help improve generalization to train deep neural networks with SNV and PID:*

- *Smaller batch size ensures better generalization.*
- *Set β_1 to be as close to 1 as possible, and set a large γ in SNV; set a large k_I in PID.*

6 VERIFYING THE GUIDELINES: EMPIRICAL EVIDENCE

We conduct extensive experiments to verify the guidelines motivated by our theoretical findings. In this section, we present our experimental results in Figure 3 and Tables 1, 2, 3.

Experimental Setup: We include different models, data, and optimizers, while sweeping across a wide range of parameters, to show the robustness of our guidelines. 'Accuracy' in the figure and tables represents *test* accuracy after training for 100 epochs. The learning curves have been flat for all experiments. Note that our objective here is not to achieve state of the art performance in these datasets. Therefore, we do not pick the setup that gives us the highest predictions and set most hyperparameters as their default values.

Figure 3: We pretrain with ImageNet and then fine-tune the model on Dogs [17] and Aircraft [27] datasets. We test ResNet18/34/50/101 (to evaluate the effect of model parameters) and VGG11 with SHB ($\nu = 1$ and $\beta = 0.9$) and PID ($k_P = -0.1$ and $k_D = 3.0$). We sweep through a large range of α from 10^{-5} to 10^{-3}.

Table 1: Left: We fit a *shallow* logistic regression on MNIST and a *deep* Preact-ResNet-110 [10] on CIFAR-10. The optimizer is SHB with $\nu = 1$ and $\beta = 0.9$. We sweep a large range of batch size and learning rate. Right: We fit a ResNet-20 [9] on CIFAR-10 with SHB ($\nu = 1$) and NAG ($\nu = \beta$), respectively. The learning rate α is fixed to be 1. We sweep β through 0 to 1..

Table 2: We fit a ResNet-20 on CIFAR-10. In the first 6 columns where we sweep α, n_b, and $\frac{n_b}{\alpha}$, the optimizer is SHB with $\nu = 1$

Table 1: Left: The effect of learning rate and batch size on shallow (LR on MNIST) vs. deep models (Preac-ResNet-110 on CIFAR-10). Right: NAG vs. SHB on small β and large β settings. 'Accuracy' represents test accuracy after training for 100 epochs. Acc_d represents the difference between NAG test accuracy and SHB test accuracy, i.e., $\text{Acc}_{\text{NAG}} - \text{Acc}_{\text{SHB}}$.

Logistic Regression on MNIST				Preact-ResNet-110 on CIFAR-10				SHB vs. NAG			
α	Accuracy	n_b	Accuracy	α	Accuracy	n_b	Accuracy	β	Acc_d	β	Acc_d
0.03	92.58%	8	91.82%	0.01	37.04%	64	85.89%	0.10	-2.22%	0.9	-0.60%
0.05	92.50%	16	91.93%	0.03	58.31%	128	83.20%	0.15	-10.54%	0.95	0.91%
0.07	92.53%	32	92.48%	0.05	66.68%	256	75.14%	0.20	-1.34%	0.99	3.37%
0.09	92.60%	64	92.60%	0.07	70.11%	300	74.08%	0.30	-10.74%	0.995	3.28%
0.20	92.54%	256	92.61%	0.1	76.60%	500	67.02%	0.40	-8.60%	0.9995	0.98%
0.30	92.54%	512	92.60%	0.2	82.80%	600	62.84%	0.45	-2.04%	0.9999	0.27%
0.40	92.35%	1024	92.42%	0.3	84.42%	700	62.33%	0.50	-0.18%	0.99995	3.41%

Table 2: ResNet-20 on CIFAR-10. We report the test accuracy with different learning rates, batch size, ratio $\frac{n_b}{\alpha}$, and (β, ν) specifications. $(\beta_{\text{rot}} = 0.99, \nu_{\text{rot}} = 0.7)$ denotes the rule of thumb specification proposed in [26]. $(\beta_{\text{opt}} = 0.999, \nu_{\text{opt}} = 0.5)$ represents a better specification justified by Theorem 3.

α	Accuracy	n_b	Accuracy	Ratio $\frac{n_b}{\alpha}$	Accuracy	$\alpha, \beta_{\text{rot}}, \nu_{\text{rot}}$	Accuracy	$\alpha, \beta_{\text{opt}}, \nu_{\text{opt}}$	Accuracy
0.010	37.54%	8	86.77%	16/0.005	71.69%	0.01	41.07%	0.01	44.67%
0.016	40.37%	16	85.89%	32/0.010	70.74%	0.014	45.87%	0.014	47.79%
0.020	41.02%	32	83.54%	64/0.020	67.02%	0.02	50.70%	0.02	53.40%
0.026	45.12%	64	76.94%	128/0.040	61.69%	0.024	49.70%	0.024	55.82%
0.030	45.57%	128	68.45%	256/0.080	54.57%	0.03	55.82%	0.03	59.05%
0.036	49.08%	256	55.08%	512/0.016	52.81%	0.036	58.38%	0.036	62.68%
0.040	50.29%	512	45.09%	800/0.025	47.09%	0.038	59.36%	0.038	62.29%
0.050	52.30%	1024	36.33%	1024/0.032	44.69%	0.040	60.08%	0.040	63.48%

Figure 3: We pretrain with ImageNet and then fine-tune the model on Dogs and Aircraft datasets. We test a list of classifiers and model architectures, larger α ensures better generalization consistently in all cases.

Table 3: PID and SNV. We report the test accuracy training ResNet-110 with CIFAR-10 using PID and SNV. We study the effect of batch size, k_I in PID control (which could be regarded as learning rate), and (γ, β_1) in SNV.

PID Control				Synthesized Nesterov Variant (SNV)					
k_I	Accuracy	n_b	Accuracy	β_1	Accuracy	γ	Accuracy	n_b	Accuracy
0.04	66.50%	64	90.25%	0.90	69.70%	0.001	48.00%	64	87.47%
0.07	73.09%	128	87.55%	0.91	72.54%	0.003	62.65%	128	85.19%
0.10	77.30%	256	77.90%	0.92	71.84%	0.005	68.74%	256	85.26%
0.13	77.80%	400	70.74%	0.93	74.20%	0.007	72.14%	300	77.31%
0.16	80.36%	600	62.72%	0.94	76.45%	0.009	75.67%	500	78.80%
0.19	81.51%	800	59.53%	0.95	79.41%	0.011	77.19%	700	71.19%
0.22	82.13%	1000	52.08%	0.96	79.86%	0.013	78.29%	900	64.64%

and $\beta = 0.999$. In the last 4 columns where we compare the rule of thumb (β, ν) specification with our suggested (β, ν) specification, the optimizer is QHM. $\beta_{\text{rot}} = 0.999$, $\nu_{\text{rot}} = 0.7$, and $\beta_{\text{opt}} = 0.999$, $\nu_{\text{opt}} = 0.7$.

Table 3: Left: We train a ResNet-110 on CIFAR-10 by PID. We set $k_P = -0.1$ and $k_D = 3.0$ when we test different k_I and n_b. Right: We train a ResNet-110 on CIFAR-10 by SNV. We set $\gamma = 0.1$ and $\beta_2 = 0.6$ when we test different β_1. We set $\beta_1 = 0.9$ and $\beta_2 = 0.6$ when we sweep γ. We set $\gamma = 0.1$, $\beta_1 = 0.9$, and $\beta_2 = 0.6$ when we test different batch sizes.

Ratio of Batch Size to Learning Rate: We conclude in the last section that smaller $\frac{n_b}{\alpha}$ ratio helps deep learning models generalize, while may have limited impact on shallow models. We report our results of training logistics regression on MNIST and Preact-ResNet-110 on CIFAR10 in Table 1. Note that the first model is a shallow model while the second model is deep and overparameterized with a sufficiently large number of parameters.

Note that with the increase of learning rate and batch size, the test accuracy of logistic regression experiences very minor fluctuation and remains practically constant around 92%. While in Preac-ResNet-110, the impact of learning rate and batch size are apparent. The test accuracy increases rapidly with larger learning rate, and drops with larger batch size. The pattern is consistent with different models and optimizers. See Table 2 for a more comprehensive study on ResNet-20 trained by CIFAR-10. Negative correlation between test accuracy and $\frac{n_b}{\alpha}$ could be observed in the first 4 columns.

Note the columns 5 and 6 in Table 2. The ratio $\frac{n_b}{\alpha}$ is fixed as 3.2×10^3. Based on the first order generalization bound in [8], the test accuracy will be approximately constant. However, we conclude from our second order bound that the accuracy will decrease with larger learning rate. Column 5 and column 6 verify our findings.

We also report the impact of batch size on other momentum-based optimizers (e.g., PID and SNV) in Table 3. The pattern that larger batch size leads to worse generalization holds for both PID and SNV.

In Figure 3, we report our results for fine-tuning. We test a list of classifiers and model architectures, larger α ensures better generalization consistently in all cases. Comparing the slopes of accuracy curve for ResNet 18, 50, and 101, we could reassure our claim that generalizability of deeper models is more positively correlated with ratio of learning rate and batch size.

NAG vs SHB: We verify our theoretical conclusion that NAG generalizes better than SHB when β is close to 1, while SHB is better when β is small. Our results are reported in the last 4 columns in Table 1.

We could observe that SHB performs much better than NAG with smaller β, typically obtaining test accuracy several percentages higher. When β is large, NAG generalizes better in general, though the improvement is minor in some cases.

Set momentum parameter β close to 1 and ν as 0.5: In the last 4 columns of Table 2, we compare our suggested hyperparameter specification $\beta_{\text{opt}} = 0.999$, $\nu_{\text{opt}} = 0.5$ with the default rule-of-thumb specification $\beta_{\text{rot}} = 0.99$, $\nu_{\text{rot}} = 0.7$ in [26].

We could observe under all learning rates, $\beta_{\text{rot}} = 0.99, \nu_{\text{rot}} = 0.7$ achieves better generalization performances than $\beta_{\text{opt}} = 0.999, \nu_{\text{opt}} = 0.5$. The improvement is non-trivial, often more than 3% higher in test accuracy, which perfectly verifies our Theorem 3.

SNV and PID: We train ResNet-110 with SNV and PID, and our results are reported in Table 3. The pattern that larger batch size leads to worse generalization holds for both PID and SNV. And the first two columns in Table 3 exhibit that larger k_I (could be regarded as learning rate in PID) improves generalization performances. The first 4 columns in SNV subtable also verify our recommended guideline: set a larger γ and β_1 in SNV training.

7 RELATED WORK

7.1 Convergence analysis for momentum methods

Many results concerning the convergence of the vanilla SGD are highlighted in [3]. Despite the widespread interest in, and use of, the stochastic momentum method, there are limited definitive theoretical convergence guarantees [5, 44].

Our analysis relies on the limited behavior of QHM iterates. [28] followed the idea of analyzing SGD with stochastic differential equations, and derived the stationary distribution of vanilla SGD and SHB. Our convergence analysis is most relevant to [6], which derived the convergence rate for QHM with constant parameters to a stability region around minimizer in the *deterministic* setting (see e.g., their Theorem 3). We extend their deterministic analysis to stochastic cases. We show how our results recover the recent convergence rate regarding NAG [2]. Furthermore, [6] was mainly purposed to minimize the *training* loss and did not consider improving generalization, which our paper focuses upon.

7.2 Generalization analysis for momemtum methods

Our work aims to theoretically and empirically justify several training heuristics regarding hyperparameters to generalize. The hyperparameter search space in state-of-the-art deep learning systems can be too high-dimensional to explore manually, especially for many interesting real-world problems, e.g., deep reinforcement learning, time series analysis, and biomedical data mining [11, 38, 39, 43]. A number of recent works [13, 15, 36] empirically report the influence of hyper-parameters, largely on batch size and learning rate, and provide practical tuning guidelines like linear scaling rule, or moving β to 1, which our paper theoretically justify.

Our generalization analysis relies on PAC-Bayesian inequalities [29, 35], and we refer readers to a comprehensive review of PAC-Bayesian learning and references therein [7]. [8, 25] proved a PAC-Bayesian bound for vanilla SGD. Our work focuses on characterizing generalization on a class of momentum schemes and covers their vanilla SGD analysis as a special case. [37] derived generalization bound for momentum schemes, but only in training from the scratch scenario. [30] proved a risk bound for transfer learning, but their bound does not connect to hyperparameters and therefore does not have practical implications on hyperparameter tuning.

8 CONCLUSIONS

In this paper, we study the convergence properties and generalization abilities of a family of momentum schemes when training deep neural networks, in both training from scratch and fine-tuning. Our analysis is unified as it covers many momentum schemes as special cases. Our theoretical findings have justified some training heuristics already known to the deep learning community, and have further inspired novel training guidelines whose effectiveness

are verified by our experiments. As momentum scheme is used so pervasively, our work could provide valuable insights for practitioners to tune hyperparameters to improve model training and generalization.

ACKNOWLEDGMENTS

The authors would like to thank the anonymous reviewers for their valuable comments and helpful suggestions. This work is supported in part by the US National Science Foundation under grants IIS-2106913, 2008208, 1955151, 1934600, 1938167. Any opinions, findings, and conclusions or recommendations expressed in this material are those of the author(s) and do not necessarily reflect the views of the National Science Foundation.

REFERENCES

[1] W. An, H. Wang, Q. Sun, J. Xu, Q. Dai, and L. Zhang. 2018. A PID Controller Approach for Stochastic Optimization of Deep Networks. In *2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition*. 8522–8531.

[2] Mahmoud Assran and M. Rabbat. 2020. On the Convergence of Nesterov's Accelerated Gradient Method in Stochastic Settings. *ArXiv* abs/2002.12414 (2020).

[3] L. Bottou, Frank E. Curtis, and J. Nocedal. 2018. Optimization Methods for Large-Scale Machine Learning. *ArXiv* abs/1606.04838 (2018).

[4] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding. *ArXiv* abs/1810.04805 (2019).

[5] Saeed Ghadimi and Guanghui Lan. 2016. Accelerated Gradient Methods for Nonconvex Nonlinear and Stochastic Programming. *Math. Program.* 156, 1–2 (March 2016), 59–99. https://doi.org/10.1007/s10107-015-0871-8

[6] Igor Gitman, Hunter Lang, Pengchuan Zhang, and Lin Xiao. 2019. Understanding the Role of Momentum in Stochastic Gradient Methods. In *Advances in Neural Information Processing Systems 32*. 9633–9643.

[7] Benjamin Guedj. 2019. A Primer on PAC-Bayesian Learning. *ArXiv* abs/1901.05353 (2019).

[8] Fengxiang He, Tongliang Liu, and Dacheng Tao. 2019. Control Batch Size and Learning Rate to Generalize Well: Theoretical and Empirical Evidence. In *Advances in Neural Information Processing Systems 32*. 1143–1152.

[9] K. He, X. Zhang, S. Ren, and J. Sun. 2016. Deep Residual Learning for Image Recognition. In *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. 770–778.

[10] Kaiming He, X. Zhang, Shaoqing Ren, and Jian Sun. 2016. Identity Mappings in Deep Residual Networks. *ArXiv* abs/1603.05027 (2016).

[11] Mengdi Huai, Jianhui Sun, Renqin Cai, Liuyi Yao, and Aidong Zhang. 2020. Malicious Attacks against Deep Reinforcement Learning Interpretations. In *Proceedings of the 26th ACM SIGKDD International Conference on Knowledge Discovery amp; Data Mining (KDD '20)*. Association for Computing Machinery, New York, NY, USA, 472–482. https://doi.org/10.1145/3394486.3403089

[12] G. Huang, Z. Liu, L. Van Der Maaten, and K. Q. Weinberger. 2017. Densely Connected Convolutional Networks. In *2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. 2261–2269.

[13] Stanisław Jastrzębski, Zac Kenton, Devansh Arpit, Nicolas Ballas, Asja Fischer, Amos Storkey, and Yoshua Bengio. 2018. Three factors influencing minima in SGD. https://openreview.net/forum?id=rJma2bZCW

[14] Yangqing Jia, Evan Shelhamer, Jeff Donahue, Sergey Karayev, Jonathan Long, Ross Girshick, Sergio Guadarrama, and Trevor Darrell. 2014. Caffe: Convolutional Architecture for Fast Feature Embedding. *arXiv preprint arXiv:1408.5093* (2014).

[15] Nitish Shirish Keskar, Dheevatsa Mudigere, Jorge Nocedal, Mikhail Smelyanskiy, and Ping Tak Peter Tang. 2016. On Large-Batch Training for Deep Learning: Generalization Gap and Sharp Minima. *CoRR* abs/1609.04836 (2016). arXiv:1609.04836 http://arxiv.org/abs/1609.04836

[16] Nitish Shirish Keskar and Richard Socher. 2017. Improving Generalization Performance by Switching from Adam to SGD. *CoRR* abs/1712.07628 (2017). arXiv:1712.07628 http://arxiv.org/abs/1712.07628

[17] Aditya Khosla, Nityananda Jayadevaprakash, Bangpeng Yao, and Li Fei-Fei. 2012. Novel Dataset for Fine-Grained Image Categorization : Stanford Dogs.

[18] Rahul Kidambi, Praneeth Netrapalli, Prateek Jain, and Sham M. Kakade. 2018. On the insufficiency of existing momentum schemes for Stochastic Optimization. *CoRR* abs/1803.05591 (2018). arXiv:1803.05591 http://arxiv.org/abs/1803.05591

[19] Laurent Lessard, Benjamin Recht, and Andrew Packard. 2014. Analysis and Design of Optimization Algorithms via Integral Quadratic Constraints. *SIAM Journal on Optimization* 26 (08 2014). https://doi.org/10.1137/15M1009597

[20] Hao Li, Pratik Chaudhari, Hao Yang, Michael Lam, Avinash Ravichandran, Rahul Bhotika, and Stefano Soatto. 2020. Rethinking the Hyperparameters for Fine-tuning. In *International Conference on Learning Representations*. https://openreview.net/forum?id=B1g8VkHFPH

[21] Hao Li, Zheng Xu, Gavin Taylor, Christoph Studer, and Tom Goldstein. 2018. Visualizing the Loss Landscape of Neural Nets. In *Proceedings of the 32nd International Conference on Neural Information Processing Systems (NIPS'18)*. Red Hook, NY, USA, 6391–6401.

[22] Xuhong Li, Yves Grandvalet, and Franck Davoine. 2018. Explicit Inductive Bias for Transfer Learning with Convolutional Networks. In *ICML*.

[23] Chaoyue Liu, Libin Zhu, and Mikhail Belkin. 2020. Toward a theory of optimization for over-parameterized systems of non-linear equations: the lessons of deep learning. arXiv:cs.LG/2003.00307

[24] Yanli Liu, Yuan Gao, and Wotao Yin. 2020. An Improved Analysis of Stochastic Gradient Descent with Momentum. arXiv:math.OC/2007.07989

[25] Ben London. 2017. A PAC-Bayesian Analysis of Randomized Learning with Application to Stochastic Gradient Descent. In *NIPS*.

[26] Jerry Ma and Denis Yarats. 2019. Quasi-hyperbolic momentum and Adam for deep learning. In *International Conference on Learning Representations*. https://openreview.net/forum?id=S1fUpoR5FQ

[27] Subhransu Maji, Esa Rahtu, Juho Kannala, Matthew B. Blaschko, and Andrea Vedaldi. 2013. Fine-Grained Visual Classification of Aircraft. *CoRR* abs/1306.5151 (2013). arXiv:1306.5151 http://arxiv.org/abs/1306.5151

[28] Stephan Mandt, Matthew D. Hoffman, and David M. Blei. 2017. Stochastic Gradient Descent as Approximate Bayesian Inference. *J. Mach. Learn. Res.* 18, 1 (Jan. 2017), 4873–4907.

[29] David A. McAllester. 1998. Some PAC-Bayesian Theorems. In *Proceedings of the Eleventh Annual Conference on Computational Learning Theory (COLT' 98)*. New York, NY, USA, 230–234. https://doi.org/10.1145/279943.279989

[30] Daniel McNamara and Maria-Florina Balcan. 2017. Risk Bounds for Transferring Representations With and Without Fine-Tuning. In *ICML*.

[31] Y. Nesterov. 1983. A method for solving the convex programming problem with convergence rate O(1/k^2).

[32] Yurii Nesterov. 2014. *Introductory Lectures on Convex Optimization: A Basic Course* (1 ed.). Springer Publishing Company, Incorporated.

[33] B.T. Polyak. 1964. Some methods of speeding up the convergence of iteration methods. *U. S. S. R. Comput. Math. and Math. Phys.* 4, 5 (1964), 1 – 17. https://doi.org/10.1016/0041-5553(64)90137-5

[34] B. T. Polyak. 1987. Introduction to Optimization.

[35] John Shawe-Taylor and Robert C. Williamson. 1997. A PAC Analysis of a Bayesian Estimator. In *Proceedings of the Tenth Annual Conference on Computational Learning Theory (COLT '97)*. New York, NY, USA, 2–9. https://doi.org/10.1145/267460.267466

[36] Samuel L. Smith, Pieter-Jan Kindermans, and Quoc V. Le. 2018. Don't Decay the Learning Rate, Increase the Batch Size. In *International Conference on Learning Representations*. https://openreview.net/forum?id=B1Yy1BxCZ

[37] Jianhui Sun, Ying Yang, Guangxu Xun, and Aidong Zhang. 2021. A Stagewise Hyperparameter Scheduler to Improve Generalization. In *Proceedings of the 27th ACM SIGKDD Conference on Knowledge Discovery amp; Data Mining*. New York, NY, USA, 1530–1540. https://doi.org/10.1145/3447548.3467287

[38] Qiuling Suo, Liuyi Yao, Guangxu Xun, Jianhui Sun, and Aidong Zhang. 2019. Recurrent Imputation for Multivariate Time Series with Missing Values. In *2019 IEEE International Conference on Healthcare Informatics, ICHI 2019, Xi'an, China, June 10-13, 2019*. IEEE, 1–3. https://doi.org/10.1109/ICHI.2019.8904638

[39] Qiuling Suo, Weida Zhong, Guangxu Xun, Jianhui Sun, Changyou Chen, and Aidong Zhang. 2020. GLIMA: Global and Local Time Series Imputation with Multi-directional Attention Learning. In *IEEE International Conference on Big Data, Big Data 2020, Atlanta, GA, USA, December 10-13, 2020*. IEEE, 798–807. https://doi.org/10.1109/BigData50022.2020.9378408

[40] Ilya Sutskever, James Martens, George Dahl, and Geoffrey Hinton. 2013. On the Importance of Initialization and Momentum in Deep Learning. In *Proceedings of the 30th International Conference on International Conference on Machine Learning - Volume 28 (ICML'13)*. III–1139–III–1147.

[41] B. Van Scoy, R. A. Freeman, and K. M. Lynch. 2018. The Fastest Known Globally Convergent First-Order Method for Minimizing Strongly Convex Functions. *IEEE Control Systems Letters* 2, 1 (2018), 49–54.

[42] Ashia C Wilson, Rebecca Roelofs, Mitchell Stern, Nati Srebro, and Benjamin Recht. 2017. The Marginal Value of Adaptive Gradient Methods in Machine Learning. In *Advances in Neural Information Processing Systems 30*. 4148–4158.

[43] Guangxu Xun, Kishlay Jha, Jianhui Sun, and Aidong Zhang. 2020. Correlation Networks for Extreme Multi-Label Text Classification. In *Proceedings of the 26th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining (KDD '20)*. New York, NY, USA, 1074–1082. https://doi.org/10.1145/3394486.3403151

[44] Yan Yan, Tianbao Yang, Zhe Li, Qihang Lin, and Yi Yang. 2018. A Unified Analysis of Stochastic Momentum Methods for Deep Learning. In *IJCAI*. 2955–2961. https://doi.org/10.24963/ijcai.2018/410

9 APPENDIX

In this section, we give the proof of Theorem 1, 2, and 3. We only keep the key proof steps and omit algebraic transformations due to page limit.

9.1 Proof of Theorem 1

We first provide the intuition of the proof. We only need to bound the norm of the right side of Equation (8) as we could see $\|\theta_k - \theta^*\|^2 = \|r_k\|^2 \leq \left\| \begin{bmatrix} d_{k-1} \\ r_k \end{bmatrix} \right\|^2$. By Gelfand's formula, we could conclude that there exists a vanishing sequence of $\{\epsilon_k\}$, such that, $\|T\|^{2k} \leq (\rho(T) + \epsilon_k)^{2k}$. We could get ε_{d} with this $\{\epsilon_k\}$ sequence.

The norm of second term in Equation (8) could be bounded by $\sum_{j=1}^{k} \|T\|^{2(k-j)} \|S\|^2 \|\xi_j\|^2$. Recall $\|S\|^2 \leq 1 - \beta^2 + \alpha^2(1-\nu\beta)^2$, and after taking expectations on both sides, $\mathbb{E}\|\xi_j\|^2 \leq \sigma^2$. Combining everything together will give us ε_{s}.

Then, we introduce one key lemma and prove Theorem 1 afterwards.

Lemma 2. *Denote the spectral radius of T as $\rho(T)$. Let the smallest and largest eigenvalues of Hessian matrix as: $\mu = \min_i \lambda_i(A)$, and $L = \max_i \lambda_i(A)$. Let $\Delta_\lambda \triangleq (1+\beta-\alpha\lambda+\alpha\nu\beta\lambda)^2 - 4\beta(1-\alpha\lambda+\alpha\lambda\nu)$. Then we have $\rho(T) = \max\{\rho_\mu, \rho_L\}$, where $\rho_\lambda = \begin{cases} \frac{1}{2}|1+\beta-\alpha\lambda+\alpha\nu\beta\lambda| + \frac{1}{2}\sqrt{\Delta_\lambda}, & \text{if } \Delta_\lambda \geq 0 \\ \sqrt{\beta(1-\alpha\lambda+\alpha\lambda\nu)}, & \text{otherwise} \end{cases}$.*

Remark 9.1. *Note that this lemma is an adapted version from [6] (see e.g., Lemmas 5, 6, 7). Here we adopt a more concise proof technique to show this lemma for completeness of this work.*

proof of Lemma 2. Recall the transition matrix T is defined as $T = \begin{bmatrix} \beta I & (1-\beta)A \\ -\alpha\nu\beta I & I - \alpha(1-\nu\beta)A \end{bmatrix}$, and T is a 2×2 block matrix with $d \times d$ blocks. As A is real and symmetric, all blocks commute with each other, since each block is an affine matrix function of A. Therefore, λ_T is an eigenvalue of T if and only if there is an eigenvalue λ_A of A, such that λ_T is an eigenvalue of the 2×2 matrix [33]: $\begin{bmatrix} \beta & (1-\beta)\lambda_A \\ -\alpha\nu\beta & 1-\alpha(1-\nu\beta)\lambda_A \end{bmatrix}$. The characteristic polynomial is: $x^2 - (1+\beta-\alpha\lambda_A+\alpha\nu\beta\lambda_A)x + \beta(1-\alpha\lambda_A+\alpha\nu\lambda_A)$. Therefore, let $\Delta_\lambda \triangleq (1+\beta-\alpha\lambda+\alpha\nu\beta\lambda)^2 - 4\beta(1-\alpha\lambda+\alpha\lambda\nu)$, and the solution of characteristic polynomial is given by ρ_{λ_A}. Let $\{\lambda_i\}_{i=1}^d$ be the set of ordered eigenvalues of A. As ρ_{λ_A} is a quasi-convex function in λ_A with fixed (α, β, ν). Therefore, ρ_{λ_A} achieves its maximum at boundary, i.e., μ and L. Therefore, we get $\rho(T) = \max\{\rho_\mu, \rho_L\}$. □

Proof of Theorem 1. Recall the dynamics of QHM: $\begin{bmatrix} d_{k-1} \\ r_k \end{bmatrix} = T^k \begin{bmatrix} d_{-1} \\ r_0 \end{bmatrix} + \sum_{j=1}^{k} T^{k-j} S \xi_j$, where d_{-1} is initialized as 0.

We extend the result for the deterministic case in [6] (see e.g. Theorem 3) to stochastic setting. Notably, in deterministic setting, the second term $\sum_{j=1}^{k} T^{k-j} S \xi_j$ is ignored. Let $\mathbb{E}$ denotes $\mathbb{E}_{\xi_1,\cdots,\xi_k}$, we have $\mathbb{E}\|\theta_k - \theta^*\|^2 = \mathbb{E}\|r_k\|^2 \leq \mathbb{E}\left\| \begin{bmatrix} d_{k-1} \\ r_k \end{bmatrix} \right\|^2 \leq \|T^k\|^2 r_0^2 + \mathbb{E}\sum_{j=1}^{k} \|T^{k-j}\|^2 \|S\|^2 \|\xi_j\|^2$.

According to Gelfand's formula: $\rho(T) = \lim_{k\to+\infty} \|T^k\|^{\frac{1}{k}}$, we would get: for any $\epsilon > 0$, there exist a $K(\epsilon)$, such that $\|T^k\|^{\frac{1}{k}} \leq \rho(T) + \epsilon$, for all $k \geq K(\epsilon)$. Let $C(\epsilon) \triangleq \max_{k<K(\epsilon)} \max\left\{1, \frac{\|T^k\|}{(\rho(T)+\epsilon)^2}\right\}$. Therefore, $\|T^k\| \leq C(\epsilon)(\rho(T)+\epsilon)^k$, and we consequently have: $\|T^k\|^2 r_0^2 + \mathbb{E}\sum_{j=1}^{k} \|T^{k-j}\|^2 \|S\|^2 \|\xi_j\|^2 \leq \varepsilon_{\mathrm{d}} + C(\epsilon)\sigma^2(1-\beta^2+\alpha^2(1-\nu\beta)^2)\sum_{j=1}^{k}(\rho(T)+\epsilon)^{2(k-j)} \leq \varepsilon_{\mathrm{d}} + \varepsilon_{\mathrm{s}}$. We obtain Theorem 1 after some straightforward algebraic transformations. □

9.2 Proof of Theorem 2

Recall the formulation of QHM, and denote the update sequence $y_k \triangleq \theta_{k+1} - \theta_k$. The updating rule is different from vanilla SGD in that $y_k \neq -\alpha\hat{g}_k$. The proof of Theorem 2 hinges on the construction of an auxiliary sequence $\{\eta_k\}_{k\in\mathbb{N}}$, such that $\eta_{k+1} - \eta_k = -\alpha\hat{g}_k$. This $\{\eta_k\}_{k\in\mathbb{N}}$ is more like vanilla SGD iterates and thus easier to deal with. We then study the property of $\{\eta_k\}_{k\in\mathbb{N}}$ and its connection to $\{\theta_k\}_{k\in\mathbb{N}}$. $\{\eta_k\}_{k\in\mathbb{N}}$ is devised as: $\eta_k = \begin{cases} \theta_k & k = 1 \\ \theta_k - \frac{\alpha\beta\nu}{1-\beta} d_{k-1} & k \geq 2 \end{cases}$, where $d_0 = 0$.

It is not difficult to verify $\eta_{k+1} - \eta_k = -\alpha\hat{g}_k$ when $k = 1$; when $k \geq 2$: $\eta_{k+1} - \eta_k = \theta_{k+1} - \frac{\alpha\beta\nu}{1-\beta} d_k - (\theta_k - \frac{\alpha\beta\nu}{1-\beta} d_{k-1}) = (-\alpha + \alpha\nu - \alpha\beta\nu)\hat{g}_k - \alpha\nu(d_k - \beta d_{k-1}) = -\alpha\hat{g}_k$.

We now study $\mathcal{R}(\eta_{k+1}) - \mathcal{R}(\eta_k)$: $\mathbb{E}_{\xi_k}[\mathcal{R}(\eta_{k+1})] \leq \mathcal{R}(\eta_k) + \mathbb{E}_{\xi_k}[< \nabla\mathcal{R}(\eta_k), \eta_{k+1} - \eta_k >] + \frac{L}{2}\mathbb{E}_{\xi_k}[\|\eta_{k+1} - \eta_k\|^2] = \mathcal{R}(\eta_k) + \mathbb{E}_{\xi_k}[< \nabla\mathcal{R}(\eta_k), -\alpha\hat{g}_k >] + \frac{L\alpha^2}{2}\mathbb{E}_{\xi_k}[\|\hat{g}_k\|^2]$.

Taking full expectation $\mathbb{E} = \mathbb{E}_{\xi_1}\mathbb{E}_{\xi_2}\ldots\mathbb{E}_{\xi_k}$ on both sides: $\mathbb{E}[\mathcal{R}(\eta_{k+1})] \leq \mathbb{E}[\mathcal{R}(\eta_k)] + \mathbb{E}[< \nabla\mathcal{R}(\eta_k), -\alpha g_k >] + \frac{L\alpha^2}{2}\mathbb{E}[\|\hat{g}_k\|^2] \leq \mathbb{E}[\mathcal{R}(\eta_k)] + \frac{L\alpha^2}{2}\mathbb{E}[\|\hat{g}_k\|^2] - \alpha\mathbb{E}[\|g_k\|^2] + \alpha\frac{c}{2}L^2\mathbb{E}[\|\eta_k - \theta_k\|^2] + \alpha\frac{1}{2c}\mathbb{E}[\|g_k\|^2]$ for $c > 0$ as any positive constant.

And we know $\eta_k - \theta_k = -\frac{\alpha\beta\nu}{1-\beta} d_{k-1}$. Thus we have: $\mathbb{E}[\mathcal{R}(\eta_{k+1})] \leq \mathbb{E}[\mathcal{R}(\eta_k)] + \alpha^3\frac{c}{2}L^2(\frac{\beta\nu}{1-\beta})^2\mathbb{E}[\|d_{k-1}\|^2] + (\alpha\frac{1}{2c} - \alpha)\mathbb{E}[\|g_k\|^2] + \frac{L\alpha^2}{2}\mathbb{E}[\|\hat{g}_k\|^2]$.

Let us make a small detour and first provide the following lemma with respect to the updating sequence $\{y_k \triangleq \theta_{k+1} - \theta_k\}_{k\in\mathbb{N}}$. Given the gradient sequence $\{\hat{g}_k\}_{k\in\mathbb{N}}$, set: $a_{k,i} = \begin{cases} 1-\beta\nu & i = k \\ \nu(1-\beta)\beta^{k-i} & i < k \end{cases}$, unroll the recursion of Equation (3) with $d_0 = 0$, and we could get $y_k = \Sigma_{i=1}^{k} a_{k,i}\hat{g}_i$. Therefore, $\mathbb{E}[y_k] = \Sigma_{i=1}^{k} a_{k,i} g_i$. We will have:

Lemma 3. *The following two inequalities hold,*

(1) The variance of QHM updating vector y_k: $\mathbb{V}[y_k] \leq \left(\frac{1-\beta}{1+\beta}\nu^2\beta^2 - \frac{1-\beta}{1+\beta}\nu^2\beta^{2k} + (1-\beta\nu)^2\right)\sigma^2$.

(2) The deviance between updating vector y_k and g_k: $\mathbb{E}[\|g_k - \frac{1}{1-\nu\beta^k}\Sigma_{i=1}^{k} a_{k,i} g_i\|^2] \leq \Sigma_{j=1}^{k-1} \frac{\nu\beta^{k-j}L^2}{1-\nu\beta^k}\left(k - j + \frac{\beta}{1-\beta}\right)\mathbb{E}[\|\theta_{j+1} - \theta_j\|^2]$.

Proof of Lemma 3. We know $\mathbb{V}[y_k] = \mathbb{E}[\|y_k - \Sigma_{i=1}^{k} a_{k,i} g_i\|^2] = \mathbb{E}[\|\Sigma_{i=1}^{k} a_{k,i}(g_i - \hat{g}_i)\|^2] \leq \Sigma_{i=1}^{k} a_{k,i}^2\sigma^2 = \left(\frac{1-\beta}{1+\beta}\nu^2\beta^2 - \frac{1-\beta}{1+\beta}\nu^2\beta^{2k} + (1-\beta\nu)^2\right)\sigma^2$, where the inequality follows from that $\{\hat{g}_k\}_{k\in\mathbb{N}}$ are independent from each other and $\mathbb{E}_{\xi_k}[\|\hat{g}_k - g_k\|^2] \leq \sigma^2$. We know $\Sigma_{i=1}^{k} a_{k,i} = 1 - \beta^k\nu$, thus we have: $\mathbb{E}[\|g_k -$

$\frac{1}{1-\nu\beta^k}\Sigma_{i=1}^k a_{k,i}g_i\|^2] = \frac{1}{(1-\nu\beta^k)^2}\mathbb{E}[\|\Sigma_{i=1}^k a_{k,i}(g_k - g_i)\|^2] \overset{(i)}{\leq} \frac{1}{(1-\nu\beta^k)^2} \times \Sigma_{i,j=1}^k a_{k,i}a_{k,j}\Big(\frac{1}{2}\mathbb{E}[\|g_k - g_j\|^2] + \frac{1}{2}\mathbb{E}[\|g_k - g_i\|^2]\Big) = \frac{1}{1-\nu\beta^k}\Sigma_{i=1}^k a_{k,i}\mathbb{E}[\|g_k - g_i\|^2] \overset{(ii)}{\leq} \frac{1}{1-\beta^k\nu}\Sigma_{i=1}^k a_{k,i}\Big((k-i)\Sigma_{j=i}^{k-1}\mathbb{E}[\|g_{j+1} - g_j\|^2]\Big) \overset{(iii)}{\leq} \frac{1}{1-\beta^k\nu}\Sigma_{i=1}^k a_{k,i}\Big((k-i)L^2\Sigma_{j=i}^{k-1}\mathbb{E}[\|\theta_{j+1} - \theta_j\|^2]\Big) = \frac{1}{1-\beta^k\nu}\Sigma_{j=1}^{k-1}\Big(\Sigma_{i=1}^j a_{k,i}(k-i)\Big)\mathbb{E}[\|\theta_{j+1} - \theta_j\|^2]L^2$, where (i) follows from Cauchy-Schwarz inequality, (ii) follows from triangle inequality, and (iii) follows from smoothness. Substitute the exact form of $a_{k,i}$, we know: $\Sigma_{i=1}^j a_{k,i}(k-i) = \nu(1-\beta)\Big(\Sigma_{i=1}^j \beta^{k-i}(k-i)\Big) \leq \nu\beta^{k-j}\Big(k - j + \frac{\beta}{1-\beta}\Big)$. Therefore, we could get the second statement. □

We know: $\mathbb{E}[\|d_{k-1}\|^2] \leq 2\mathbb{E}[\|d_{k-1} - (1-\beta)\Sigma_{i=1}^{k-1}\beta^{k-1-i}g_i\|^2] + 2\mathbb{E}[\|(1-\beta)\Sigma_{i=1}^{k-1}\beta^{k-1-i}g_i\|^2] \leq 2\frac{1-\beta}{1+\beta}\sigma^2 + 2\mathbb{E}[\|(1-\beta)\Sigma_{i=1}^{k-1}\beta^{k-1-i}g_i\|^2]$, where it follows from Lemma 1 in [24]. And we also have: $\mathbb{E}[\|\frac{1-\beta}{1-\beta^{k-1}}\Sigma_{i=1}^{k-1}\beta^{k-1-i}g_i\|^2] \leq 2\mathbb{E}[\|\frac{1-\beta}{1-\beta^{k-1}}\Sigma_{i=1}^{k-1}\beta^{k-1-i}g_i - g_k\|^2] + 2\mathbb{E}[\|g_k\|^2]\mathbb{E}[\|\hat{g}_k\|^2] \leq \sigma^2 + \mathbb{E}[\|g_k\|^2]$.

Substitute the above inequalities back to the inequality prior to Lemma 3: $\mathbb{E}[\mathcal{R}(\eta_{k+1})] \leq \mathbb{E}[\mathcal{R}(\eta_k)] + \Big(-\alpha + \alpha\frac{1}{2c} + 2\alpha^3cL^2(\frac{\beta\nu}{1-\beta})^2(1-\beta^{k-1})^2 + \frac{L\alpha^2}{2}\Big)\mathbb{E}[\|g_k\|^2] + \Big(\alpha^3cL^2(\frac{\beta\nu}{1-\beta})^2\frac{1-\beta}{1+\beta} + \frac{L\alpha^2}{2}\Big)\sigma^2 + 2\alpha^3cL^2(\frac{\beta\nu}{1-\beta})^2(1-\beta^{k-1})^2 \times \mathbb{E}[\|\frac{1-\beta}{1-\beta^{k-1}}\Sigma_{i=1}^{k-1}\beta^{k-1-i}g_i - g_k\|^2]$. We know: $\mathbb{E}[\|\frac{1-\beta}{1-\beta^{k-1}}\Sigma_{i=1}^{k-1}\beta^{k-1-i}g_i - g_k\|^2] = \frac{1}{\beta^2}(\frac{1-\beta^k}{1-\beta^{k-1}})^2\mathbb{E}[\|\frac{1-\beta}{1-\beta^k}\Sigma_{i=1}^k\beta^{k-i}g_i - g_k\|^2]$. Let $c = \frac{1-\beta}{2L\alpha}$, we have $\mathbb{E}[\mathcal{R}(\eta_{k+1})] \leq \mathbb{E}[\mathcal{R}(\eta_k)] + \Big(\frac{\beta^2\nu^2}{2(1+\beta)} + \frac{1}{2}\Big)L\alpha^2\sigma^2 + L\alpha^2\nu^2\frac{(1-\beta^k)^2}{1-\beta}\mathbb{E}[\|\frac{1-\beta}{1-\beta^k}\Sigma_{i=1}^k\beta^{k-i}g_i - g_k\|^2] + \Big(-\alpha + \frac{1}{2}L\alpha^2 + \frac{1+\beta^2\nu^2}{1-\beta}L\alpha^2\Big)\mathbb{E}[\|g_k\|^2]$. We study the following sequence: $L_k \triangleq \mathcal{R}(\eta_k) - \mathcal{R}^* + \Sigma_{i=1}^{k-1}q_i\|\theta_{k+1-i} - \theta_{k-i}\|^2$ following the idea from [24], where q_i are constants to be determined: $\mathbb{E}[L_{k+1} - L_k] \leq q_1\mathbb{E}[\|\theta_{k+1} - \theta_k\|^2] + \Sigma_{i=1}^{k-1}(q_{i+1} - q_i)\mathbb{E}[\|\theta_{k+1-i} - \theta_{k-i}\|^2]\Big(-\alpha + \alpha\frac{1}{2c} + 2\alpha^3cL^2(\frac{\beta\nu}{1-\beta})^2 + \frac{L\alpha^2}{2}\Big)\mathbb{E}[\|g_k\|^2] + \Big(\alpha^3cL^2(\frac{\beta\nu}{1-\beta})^2\frac{1-\beta}{1+\beta} + \frac{L\alpha^2}{2}\Big)\sigma^2 + 2\alpha^3cL^2(\frac{\nu}{1-\beta})^2(1-\beta^k)^2 \times \mathbb{E}[\|\frac{1-\beta}{1-\beta^k}\Sigma_{i=1}^k\beta^{k-i}g_i - g_k\|^2]$.

Let us look at $q_1\mathbb{E}[\|\theta_{k+1} - \theta_k\|^2]$ in detail: $q_1\mathbb{E}[\|\theta_{k+1} - \theta_k\|^2] = q_1\alpha^2\mathbb{E}[\|y_k\|^2] \leq 2q_1\alpha^2\Big(\frac{1-\beta}{1+\beta}\nu^2\beta^2 - \frac{1-\beta}{1+\beta}\nu^2\beta^{2k} + (1-\beta\nu)^2\Big)\sigma^2 + 2q_1\alpha^2(1-\nu\beta^k)^2\mathbb{E}[\|\frac{1}{1-\nu\beta^k}\Sigma_{i=1}^k a_{k,i}g_i\|^2]$.

We know: $\mathbb{E}[\|\frac{1}{1-\nu\beta^k}\Sigma_{i=1}^k a_{k,i}g_i\|^2] \leq 2\mathbb{E}[\|\frac{1}{1-\nu\beta^k}\Sigma_{i=1}^k a_{k,i}g_i - g_k\|^2] + 2\mathbb{E}[\|g_k\|^2]$. Therefore, we have $q_1\mathbb{E}[\|\theta_{k+1} - \theta_k\|^2] \leq 4q_1\alpha^2(1-\nu\beta^k)^2\mathbb{E}[\|g_k\|^2] + 2q_1\alpha^2\Big(\frac{1-\beta}{1+\beta}\nu^2\beta^2 - \frac{1-\beta}{1+\beta}\nu^2\beta^{2k} + (1-\beta\nu)^2\Big)\sigma^2 + 4q_1\alpha^2(1-\nu\beta^k)^2\mathbb{E}[\|\frac{1}{1-\nu\beta^k}\Sigma_{i=1}^k a_{k,i}g_i - g_k\|^2]$.

We study $\mathbb{I} \triangleq \Sigma_{i=1}^{k-1}(q_{i+1} - q_i)\mathbb{E}[\|\theta_{k+1-i} - \theta_{k-i}\|^2] + 4q_1\alpha^2(1-\nu\beta^k)^2 \times \Sigma_{j=1}^{k-1}\frac{\nu\beta^{k-j}L^2}{1-\nu\beta^k}\Big(k-j+\frac{\beta}{1-\beta}\Big)\mathbb{E}[\|\theta_{j+1} - \theta_j\|^2] + 2\alpha^3cL^2(\frac{\nu}{1-\beta})^2(1-\beta^k)^2 \times \Sigma_{j=1}^{k-1}\frac{\beta^{k-j}L^2}{1-\beta^k}\Big(k-j+\frac{\beta}{1-\beta}\Big)\mathbb{E}[\|\theta_{j+1} - \theta_j\|^2] \leq \Sigma_{i=1}^{k-1}(q_{i+1} - q_i)\mathbb{E}[\|\theta_{k+1-i} - \theta_{k-i}\|^2] + 4q_1\alpha^2(1-\nu\beta^k)^2 \times \Sigma_{i=1}^{k-1}\frac{\beta^iL^2}{1-\beta^k}\Big(i + \frac{\beta}{1-\beta}\Big)\mathbb{E}[\|\theta_{k+1-i} - \theta_{k-i}\|^2] + 2\alpha^3cL^2(\frac{\nu}{1-\beta})^2(1-\beta^k)^2 \times \Sigma_{i=1}^{k-1}\frac{\beta^iL^2}{1-\beta^k}\Big(i + \frac{\beta}{1-\beta}\Big)\mathbb{E}[\|\theta_{k-i+1} - \theta_{k-i}\|^2]$

In order to let $\mathbb{I}$ be non-positive, we need to have for all $i \geq 1$: $q_{i+1} \leq q_i - \Big(4q_1\alpha^2(1-\nu\beta^k)^2 + 2\alpha^3cL^2(\frac{\nu}{1-\beta})^2(1-\beta^k)^2\Big)\frac{\beta^iL^2}{1-\beta^k}\Big(i + \frac{\beta}{1-\beta}\Big)$. It suffices to have for all $i \geq 1$: $q_{i+1} = q_i - \Big(4q_1\alpha^2\frac{1}{\nu(1-\beta)} + 2\alpha^3cL^2(\frac{\nu}{1-\beta})^2\Big)\beta^iL^2\Big(i + \frac{\beta}{1-\beta}\Big)$.

Therefore, by some algebraic transformations, in order for $q_i > 0$ for all $i \geq 1$, q_1 could be set as $q_1 = \Big(4q_1\alpha^2\frac{1}{\nu(1-\beta)} + 2\alpha^3cL^2(\frac{\nu}{1-\beta})^2\Big)\Sigma_{i=1}^\infty\beta^iL^2\Big(i + \frac{\beta}{1-\beta}\Big) = \frac{2\alpha^3cL^4\frac{\beta+\beta^2}{(1-\beta)^4}\nu^2}{1-4\alpha^2\frac{\beta+\beta^2}{\nu(1-\beta)^3}L^2}$. Combining everything together, and we could determine $\{q_k\}_{k\in\mathbb{N}}$ such that $\mathbb{I}$ is non-positive.

We now have: $\mathbb{E}[L_{k+1} - L_k] \leq -Q_1\mathbb{E}[\|g_k\|^2] + Q_2\sigma^2$, where $Q_1 \triangleq \alpha - \alpha\frac{1}{2c} - 2\alpha^3cL^2(\frac{\beta\nu}{1-\beta})^2 - \frac{L\alpha^2}{2} - 4q_1\alpha^2(1-\nu\beta^k)^2$ and $Q_2 \triangleq \Big(\alpha^3cL^2(\frac{\beta\nu}{1-\beta})^2\frac{1-\beta}{1+\beta} + \frac{L\alpha^2}{2}\Big) + 2q_1\alpha^2\Big(\frac{1-\beta}{1+\beta}\nu^2\beta^2 - \frac{1-\beta}{1+\beta}\nu^2\beta^{2k} + (1-\beta\nu)^2\Big)$

Omitting the tedious transformations, we could get, if $\alpha \leq \min\{\frac{1-\beta}{(3+2\beta^2\nu^2+2\nu^4)L}, \frac{\nu(1-\beta)\sqrt{1-\beta}}{2\sqrt{2}\sqrt{\beta+\beta^2}L}\}$, we have: $\frac{1}{k}\Sigma_{i=1}^k\mathbb{E}[\|g_i\|^2] \leq \frac{2\Big(\mathcal{R}(\theta_1)-\mathcal{R}^*\Big)}{k\alpha} + \Big(1 + \frac{\beta^2\nu^2}{1+\beta} + \frac{1-\beta}{1+\beta}\beta^2\nu^6 + (1-\beta\nu)^2\nu^4\Big)L\alpha\sigma^2$.

9.3 Proof of Theorem 3

We introduce the following lemma and then prove Theorem 3.

Lemma 4 ([29]). *Let $KL(Q||P)$ as the KL divergence between two distributions Q and P. For any positive real $\delta \in (0,1)$, with probability at least $1-\delta$ over a sample of size N, we have the following inequality for all distributions Q: $\mathcal{R}(Q) \leq \hat{\mathcal{R}}(Q) + \sqrt{\frac{KL(Q||P)+\log\frac{1}{\delta}+\log N+2}{2N-1}}$.*

Lemma 1 can be adapted from Theorem 5 in [6] with some algebraic transformations. We omit the detailed proof due to space limit. As we are mainly concerned about how α, β, ν affect the trend of generalization bound, we ignore higher order terms for now. The experiments have shown that our approximations are satisfactory.

proof of Theorem 3. With Lemma 4 and Lemma 1, we are ready to prove Theorem 3. Recall the density of prior and posterior distributions are $f_P = \frac{1}{\sqrt{2\pi\det(\lambda_0 I_d)}}\exp\Big\{-\frac{1}{2}(\theta-\theta_0)^T(\lambda_0 I_d)^{-1}(\theta-\theta_0)\Big\}$ and $f_Q = \frac{1}{\sqrt{2\pi\det(\Sigma_\theta)}}\exp\Big\{-\frac{1}{2}\theta^T\Sigma_\theta^{-1}\theta\Big\}$, respectively. We calculate their $KL(Q||P)$ as follows: $KL(Q||P) = \int\Big(\frac{1}{2}\log\frac{|\lambda_0 I_d|}{|\Sigma_\theta|} - \frac{1}{2}\theta^T\Sigma_\theta^{-1}\theta + \frac{1}{2}(\theta-\theta_0)^T(\lambda_0 I_d)^{-1}(\theta-\theta_0)\Big)f_Q(\theta)d\theta = \frac{1}{2}\Big\{\mathrm{tr}((\lambda_0 I_d)^{-1}\Sigma_\theta) + \theta_0^T(\lambda_0 I_d)^{-1}\theta_0 - d + \log\frac{|\lambda_0 I_d|}{|\Sigma_\theta|}\Big\} = \frac{1}{2\lambda_0}\theta_0^T\theta_0 - \frac{d}{2} + \frac{d}{2}\log\lambda_0 + \frac{1}{2\lambda_0}\mathrm{tr}(\Sigma_\theta) - \frac{1}{2}\log|\Sigma_\theta|$. A direct application of the determinant and trace from Lemma 1 completes our proof for training from scratch. The scenario for fine-tuning follows the exact same proof logic. □