THE IMPACT OF THE MINI-BATCH SIZE ON THE DYNAMICS OF SGD: VARIANCE AND BEYOND*

ANONYMOUS AUTHORS

Abstract. We investigate the mini-batch stochastic gradient descent (SGD) dynamics under deep polynomially-activated networks and general feed-forward neural networks by representing the stochastic gradient (SG) estimators using only the initial weights and mini-batch size – a novel and under-explored approach. For polynomially-activated networks, we derive recursive relationships between the norms of gradients and weight matrices across consecutive time steps. Additionally, we demonstrate that in each iteration, the norm of the gradient is a polynomial function of the reciprocal of the mini-batch size and decreases with increasing mini-batch size. In the more general case, we show that, the norm of the gradient in a general feed-forward network can be arbitrarily well approximated by some polynomially-activated networks at any given time step.

These results theoretically back the widely accepted intuition that smaller batch sizes yield larger variance of the SG estimators and lower loss function values. The proof techniques exhibit explicit connections between various general functions of SG estimators and initial weights, facilitating further research on SGD dynamics. We provide empirical insights into our findings across multiple datasets and commonly used deep network structures, and discuss potential extensions of these approaches to study the generalization capabilities of deep learning models. Updated abstract a bit. please also check the keywords below.

Key words. Stochastic Gradient Descent, Polynomially-activated Neural Networks, Variance of SG estimators

AMS subject classifications. 68Q25, 68R10, 68U05

1. Introduction.

Deep learning models have achieved great success in a variety of tasks including natural language processing, computer vision, and reinforcement learning [9]. Despite their practical success, there are only limited studies of the theoretical properties of deep learning; see survey papers [39, 8] and references therein. The general problem underlying deep learning models is to optimize (minimize) a loss function, defined by the deviation of model predictions on data samples from the corresponding true labels. The prevailing method to train deep learning models is the mini-batch stochastic gradient descent algorithm and its variants [4, 5]. SGD updates model parameters by calculating a stochastic approximation of the full gradient of the loss function, based on a random selected subset of the training samples called a mini-batch.

Although SGD can converge to the minimum of a convex function [6], deep neural networks are strongly non-convex. Thus, the success of SGD in neural network training, especially the dynamics of SGD, becomes an interesting question. Some researchers approximate the dynamics of SGD by a continuous-time dynamic system [26, 25, 28, 17]. Another line of research [27, 7, 1] show that the dynamics of SGD in training over-parameterized neural networks are similar to training a linear model. However, these statements are approximate in nature and do not provide explicit formulas for calculating any specific quantities during SGD training. The mini-batch size is also a key factor deciding the dynamics of SGD. Some research focuses on how to choose an optimal mini-batch size based on different criteria [38, 11]. However, these works make strong assumptions on the loss function properties (strong or point or quasi convexity, or constant variance near stationary points) or about the formulation of the SGD algorithm (continuous time interpretation by means of differential equations). The theoretical results regarding the relationship between the mini-batch size and

*Submitted to the editors on DATE.

the variance (and other performances, like loss and generalization ability) of the SGD algorithm applied to general machine learning models are still missing.

Besides, it is well-accepted that selecting a large mini-batch size reduces the training time of deep learning models, as computation on large mini-batches can be better parallelized on processing units. For example, Goyal et al. [12] scale ResNet-50 [13] from a mini-batch size of 256 images and training time of 29 hours, to a larger mini-batch size of 8,192 images. Their training achieves the same level of accuracy while reducing the training time to one hour. However, noted by many researchers, larger mini-batch sizes suffer from a worse generalization ability [22, 19]. Therefore, many efforts have been made to develop specialized training procedures that achieve good generalization using large mini-batch sizes [16, 12]. Smaller batch sizes have the advantage of allegedly offering better generalization (at the expense of a higher training time). We hypothesize that, given the same initial point, smaller sizes lead to lower training loss and, unfortunately, decrease stability of the algorithm on average. The latter follows from the fact that the smaller is the batch size, more stochasticity and volatility is introduced. After all, if the batch size equals to the number of samples, there is no stochasticity in the algorithm. To this end, we conjecture that the variance of the gradient in each iteration is a decreasing function of the mini-batch size. We partially prove this conjecture in this work.

In this paper, we study the dynamics of SGD by representing related quantities only using the mini-batch size, initial points and learning rates, which are available before training. This is different from previous literature which analyzes SGD by focusing on one-step properties. In fact, the dynamics of SGD are not comparable if we merely consider the one-step behavior, as the model parameters change iteration by iteration. We develop general frameworks for studying the dynamics of mini-batch SGD in deep polynomially-activated neural networks. These frameworks offer explicit and recursive relationships of various general forms that encompass multiple aspects of SGD dynamics. Additionally, we investigate the dynamics of general feed-forward networks by approximating them with polynomially-activated networks.

As an application of our frameworks, we validate the hypothesis regarding variance in both deep-polynomially activated network and general feed-forward settings. We demonstrate that variance is a polynomial function of the reciprocal of the mini-batch size and decreases if the mini-batch size surpasses a specific threshold (subsequent experiments show that this threshold can be as low as 1). The increased variance with smaller mini-batch sizes should intuitively result in convergence to lower training loss values and ultimately better prediction and generalization capabilities (although these relationships remain to be confirmed analytically, we offer empirical evidence supporting their validity).

The major contributions of this paper are as follows.

(i) For deep polynomially-activated neural network under a teacher-student setting, we build a framework to recursively calculate the trace of any product of the SG estimators, weight matrices, and other constant matrices at time step t by using the variables at time step $t-1$ (Theorems 3.1 and 3.2). This explicit relationship can be used to derive the expected value of the product of the weight matrices and SG estimators as a polynomial in $1/b$ with coefficients a sum of products of the initial weights (Theorem 3.3). As a special case, the variance of the SG estimator is a polynomial in $1/b$ without the constant term (Theorem 3.4) and therefore it is a decreasing function of b when b is large enough (Theorem 3.5). The results and proof techniques can be extended in an approximate sense to deep networks with general non-linear activation functions (Section 3.2). As a comparison, other papers that study theoretical properties of two-layer networks either fix one layer of the network, or assume the over-parameterized property of the model and they study convergence, while our paper makes no such assumptions on the model capacity. The proof also reveals the structure of the coefficients of the polynomial, and thus it serves as a tool for future work on proving other properties of the SG estimators and weight matrices.

(ii) The proofs are involved and require several key ideas. The main one is to show a more general result than it is necessary in order to carry out the induction on time step t. New concepts and definitions are introduced in order to handle the more general case. Along the way we show a result of general interest establishing expectation of the product of quadratic terms of samples with general distribution intertwined with constant matrices.

(iii) We verify the theoretical results regarding the decreasing property of variance on various datasets and provide a further understanding. We also empirically show that the results extend to other widely used network structures and hold for all choices of the mini-batch sizes. We also empirically verify that, on average, in each iteration the loss function value and the generalization ability (measured by the gap between accuracy on the training and test sets) are all decreasing functions of the mini-batch size.

In conclusion, we study the dynamics of SGD under deep polynomially-activated network and general feed-forward networks settings by building frameworks that can recursively and explicitly calculate general products and sums of the SG estimators and weights matrices between consecutive iterations. As an application of the frameworks, we focus on representing the variance of the SG estimators by the mini-batch size, initial weights and other constant variables, and therefore prove the decreasing property of the variance of the SG estimators. The proof techniques can also be used to derive other properties of the SGD dynamics in regard to the mini-batch size and initial weights. To the best of authors' knowledge, the work is the first one to theoretically and explicitly study the important quantities of SGD at iteration t only using the initial weights and mini-batch size, under mild assumptions on the network and the loss function. We support our theoretical results by experiments. We further experiment on other state-of-the-art deep learning models and datasets to empirically show the validity of the conjectures about the impact of mini-batch size on average loss, average accuracy and the generalization ability of a model.

The rest of the manuscript is structured as follows. In Section 2 we review the literature while in Section 3 we present a general framework on how to recursively represent some functions of the SG estimators by initial weights, under different models including deep polynomially-activated networks and general neural networks. We also provide applications of the presented framework in Section 3. Section 4 presents the experiments that verify our theorems and provide further insights into the impact of the mini-batch sizes on SGD dynamics. The proofs of the theorems and other technical details are available in Appendix A.

2. Literature Review.

Stochastic gradient descent type methods are broadly used in machine learning [3, 21, 5]. The performance of SGD highly relies on the choice of the mini-batch size. It has been widely observed that choosing a large mini-batch size to train deep neural networks appears to deteriorate generalization [22]. This phenomenon exists even if the models are trained without any budget or limits, until the loss function value ceases to improve [19]. One explanation for this phenomenon is that large mini-batch SGD produces "sharp" minima that generalize worse [15, 19]. Specialized training procedures to achieve good performance with large mini-batch sizes have also been proposed [16, 12].

It is well-known that SGD has a slow asymptotic rate of convergence due to its inherent variance [18]. Variants of SGD that can reduce the variance of the SG estimator, which yield faster convergence, have also been suggested. The use of the information of full gradients to provide variance control for stochastic gradients is addressed in [18, 34, 36]. The works in [23, 24, 35] further improve the efficiency and complexity of the algorithm by carefully controlling the variance.

There is prior work focusing on studying the dynamics of SGD. Neelakantan et al. propose to add isotropic white noise to the full gradient to study the "structured" variance [31]. The works in [25, 28, 17] connect SGD with stochastic differential equations to explain the property of converged minima and generalization ability of the model. Smith et al. propose an "optimal" mini-batch size which maximizes the test set accuracy by a Bayesian approach [38]. The Stochastic Gradient Langevin Dynamics (SGLD, a variant of SGD) algorithm for non-convex optimization is studied in [43, 30].

In most of the prior work about the convergence of SGD, it is assumed that the variance of SG estimators is upper-bounded by a linear function of the norm of the full gradient, e.g. Assumption 4.3 in [5]. Gower et al. [11] give more precise bounds of the variance under different sampling methods and Khaled et al. [20] extend them to smooth non-convex regime. These bounds are still dependent on the model parameters at the corresponding iteration. To the best of the authors' knowledge, there is no existing result which represents SG estimators only using the initial weights and the mini-batch size. This paper partially solves this problem.

3. Analysis. Mini-batch SGD is a lighter-weight version of gradient descent. Suppose that we are given a loss function $L(w)$ where w is the collection (vector, matrix, or tensor) of all model parameters. At each iteration t, instead of computing the full gradient $\nabla_w L(w_t)$, SGD randomly samples a mini-batch set $\mathcal{B}_t$ that consists of $b = |\mathcal{B}_t|$ training instances and sets $w_{t+1} \leftarrow w_t - \alpha_t \nabla_w L_{\mathcal{B}_t}(w_t)$, where the positive scalar α_t is the learning rate (or step size) and $\nabla_w L_{\mathcal{B}_t}(w_t)$ denotes the SG estimator based on mini-batch $\mathcal{B}_t$.

An important property of the SG estimator $\nabla_w L_{\mathcal{B}_t}(w_t)$ is that it is an unbiased estimator, i.e. $\mathbb{E}\left[\nabla_w L_{\mathcal{B}_t}(w_t)\right] = \nabla_w L(w_t)$, where the expectation is taken over all possible choices of mini-batch $\mathcal{B}_t$. However, it is unclear what is the value of[1]

$$\mathsf{var}\left(\nabla_w L_{\mathcal{B}_t}(w_t)\right) := \mathbb{E}\left\|\nabla_w L_{\mathcal{B}_t}(w_t)\right\|^2 - \left\|\mathbb{E}\nabla_w L_{\mathcal{B}_t}(w_t)\right\|^2.$$

Intuitively, we should have $\mathsf{var}\left(\nabla_w L_{\mathcal{B}_t}(w_t)\right) \propto \frac{n^2}{b}\mathsf{var}\left(\nabla_w L(w_t)\right)$, where n is the number of training samples and stochasticity on the right-hand side comes from mini-batch samples behind w_t [38, 11]. However, even the quantities $\nabla_w L(w_t)$ and $\mathsf{var}\left(\nabla_w L(w_t)\right)$ are still challenging to compute as we do not have direct formulas of their precise values. Besides, as we choose different b's, their values are not comparable as we end up with different w_t's.

A plausible idea to address these issues is to represent $\mathbb{E}\left[\nabla_w L_{\mathcal{B}_t}(w_t)\right]$ and $\mathsf{var}(\nabla_w L_{\mathcal{B}_t}(w_t))$ only using the fixed and known quantities w_0, b, t, and α_t. In this way, we can further discover the properties, like decreasing with respect to b, of $\mathbb{E}\left[\nabla_w L_{\mathcal{B}_t}(w_t)\right]$ and $\mathsf{var}\left(\nabla_w L_{\mathcal{B}_t}(w_t)\right)$. The biggest challenge is how to connect the quantities in iteration t with those of iteration 0. This is similar to discovering the properties of a stochastic

[1]Note that this definition is different from the variance of a vector, i.e., the covariance matrix. This "scalar" variance is a common practice in the field of optimization (e.g. equation (4.6) in [5]).

differential equation at time t given only the dynamics of the stochastic differential equation and the initial point. Anonymous Authors [2] present explicit formulas for calculating any norm of the linear combination of sample-wise gradients in the linear regression setting. Although this setup is simpler than that of neural networks, it necessitates non-trivial mathematical proofs to establish the connection between SG estimators at iteration t using only information available at iteration 0.

In this section, we address these questions by recursively representing some general forms of SG estimators under two settings: deep polynomially-activated networks and general feed-forward neural networks. In Section 3.1, under a deep polynomially-activated network with teacher-student setting, we provide explicit formulas for calculating any trace of the mixed product of weight matrices and SG estimators. With this tool, we further show that these traces are polynomials in $1/b$ with finite degree and that $\mathsf{var}\left(\nabla_w L_{\mathcal{B}_t}(w_t)\right)$ is a decreasing function of the mini-batch size $b > b_0$ for some constant b_0. In Section 3.2, we extend the results to general deep neural networks with mild assumptions on the activation functions in an approximate sense.

For a random matrix M, we define $\mathsf{var}\,(M) := \mathbb{E}\left\|\mathrm{vec}(M)\right\|^2 - \left\|\mathbb{E}\mathrm{vec}(M)\right\|^2$ where $\mathrm{vec}(M)$ denotes the vectorization of matrix M. We denote $[m:n] := \{m, m+1, \ldots, n\}$ if $m \leqslant n$, and $\varnothing$ otherwise. We use $[n] := [1:n]$ as an abbreviation. For clarity, we use the superscript b to distinguish the variables with different choices of the mini-batch size b. In each iteration t, we use $\mathcal{B}_t^b$ to denote the batch of samples (or sample indices) to calculate the stochastic gradient. We denote by $\mathcal{F}_t^b$ the filtration of information before calculating the stochastic gradient in the t-th iteration, i.e. $\mathcal{F}_t^b := \left\{w_0, w_1^b, \ldots, w_t^b, \mathcal{B}_0^b, \ldots, \mathcal{B}_{t-1}^b\right\}$. We use $\bigotimes_{i\in[n]} A_i$ to denote the Kronecker product of matrices $A_1, \ldots, A_n$.-

3.1. Deep Networks with Polynomial Activation Functions. In this section, we investigate the dynamics of SGD on deep networks utilizing a polynomial activation function. We present the informal theorems in this section and reserve the complete versions for the Appendix. Additionally, we provide a comprehensive proof of the two-layer linear network (which corresponds to a polynomial activation of degree one) in the Appendix, along with the necessary additions to extend the proof to the multi-layer polynomial case.

Given a distribution $\mathcal{D}$ in $\mathbb{R}^p$, we consider the population loss

$$\mathcal{L}(w) = \mathbb{E}_{x\sim\mathcal{D}}\left[\frac{1}{2}\left\|W_H\sigma\left(W_{H-1}\sigma\left(\cdots\sigma\left(W_1x\right)\right)\right) - W_H^*\sigma\left(W_{H-1}^*\sigma\left(\cdots\sigma\left(W_1^*x\right)\right)\right)\right\|^2\right] \tag{3.1}$$

under the teacher-student learning framework [14] with $w = (W_1, W_2, \cdots, W_H)$ a set of weight matrices. Here $W_k \in \mathbb{R}^{p_k\times p_{k-1}}, k \in [H], p_0 = p$ are parameter matrices of the student network, $W_k^*, k \in [H]$ are the fixed ground-truth parameters of the teacher network, and $\sigma(\cdot)$ is a polynomial with degree D. We use online SGD to minimize the population loss $\mathcal{L}(w)$. Formally, we first choose a mini-batch size b and initial weight matrices $\{W_{0,k}, k \in [H]\}$; in each iteration t, we independently draw a mini-batch $\mathcal{B}_t^b := \left\{x_{t,i}^b : i \in [b]\right\}$ of b samples from $\mathcal{D}$ and update the weight matrices by $W_{t+1,k}^b = W_{t,k}^b - \alpha_t g_{t,k}^b$, where

$$g_{t,k}^b := \frac{1}{b}\sum_{i=1}^b \nabla_{W_{t,k}^b}\left(\frac{1}{2}\left\|W_{t,H}^b\sigma\left(W_{t,H-1}^b\sigma\left(\cdots\sigma\left(W_{t,1}^bx_{t,i}^b\right)\right)\right) - W_H^*\sigma\left(W_{H-1}^*\sigma\left(\cdots\sigma\left(W_1^*x_{t,i}^b\right)\right)\right)\right\|^2\right).$$

For a multi-set of matrices $\mathcal{M} = \{M_1, \ldots, M_n\}$, we use $\deg\left(A;\mathcal{M}\right)$ to denote the number of appearances of matrix A and its transpose A^T in $\mathcal{M}$. Mathematically, we have $\deg(A;\mathcal{M}) := \sum_{i\in[n]}\left(\mathbb{I}\{A=M_i\}+\mathbb{I}\{A^T=M_i\}\right)$. We further denote $\deg(\mathcal{A};\mathcal{M}) := \sum_{A\in\mathcal{A}}\deg(A;\mathcal{M})$ for any set of matrices $\mathcal{A}$. We denote $W_t^b := \left\{W_{t,k}^b, k\in[H]\right\}$, $W_{:t}^b = \bigcup_{s\in[0:t]} W_s^b$, $G_t^b := \left\{g_{t,k}^b, k\in[H]\right\}$, $G_{:t}^b = \bigcup_{s\in[0:t]} G_s^b$, and $W^* := \{W_k^*, k\in[H]\}$. We use $\mathcal{C}$ to denote the infinite set of all non-random matrices given $\mathcal{F}_0$.[2]

3.1.1. Dynamics: Connecting Generalized Products Step by Step. As pointed out in the Section 1, the difficulty of studying the dynamics of SGD is how to connect the quantities in iteration t with fixed variables, like the initial weights $W_{0,k}^b$ and mini-batch size b. We overcome this challenge by carefully building the connection between (i) $g_{t,k}^b$ and $W_{t,k}^b, k\in[H]$; (ii) $W_{t,k}^b$ and $g_{t-1,k}^b, k\in[H]$. The following two theorems address these two questions by considering a term of mixed product of $W_{t,k}^b$ and $g_{t,k}^b$, respectively.

THEOREM 3.1. *Let $\mathcal{M} := \{M_{i,j} : i\in[0:I], j\in[J]\}$ be a multi-set of matrices such that each $M_{i,j}$ or its transpose only takes value in $W_{:t}^b\bigcup G_{:t}^b\bigcup\mathcal{C}$ and $\deg\left(G_t^b;\mathcal{M}\right)=d$. Then there exist constants I', J', L_s independent of b and a multi-set of matrices $\mathcal{Q}=\{Q_{l,s,i,j}, l\in[L_s], i\in[0:I'], j\in[J'], s\in[0:d]\}$ such that*

$$\mathbb{E}\left[\operatorname{tr}\left(C\left(\bigotimes_{i\in[I]}\prod_{j\in[J]}M_{i,j}\right)\right)\prod_{j\in[J]}M_{0,j}\middle|\mathcal{F}_t^b\right]=\sum_{s=0}^{d}Q_s\frac{1}{b^s}\tag{3.2}$$

where $Q_s = \sum_{l\in[L_s]} c_{l,s}\operatorname{tr}\left(C_{l,s}\left(\bigotimes_{i\in[I']}\prod_{j\in[J']}Q_{l,s,i,j}\right)\right)\prod_{j\in[J']}Q_{l,s,0,j}$, $s\in[0:d]$, $c_{l,s}$ is a constant, $C, C_{l,s}\in\mathcal{C}$ are constant matrices, and $Q_{l,s,i,j}\in W_{:t}^b\bigcup G_{:t-1}^b\bigcup\mathcal{C}$.

Note that the randomness of $\operatorname{tr}\left(C\left(\bigotimes_{i\in[I]}\prod_{j\in[J]}M_{i,j}\right)\right)\prod_{j\in[J]}M_{0,j}$ in (3.2) only comes from $G_t^b=\left\{g_{t,k}^b, k\in[H]\right\}$ while conditioning on $\mathcal{F}_t^b$. Together with the fact that each $Q_{l,s,i,j}$ involves only $W_{:t}^b\bigcup G_{:t-1}^b\bigcup\mathcal{C}$, Theorem 3.1 enables the induction step from $g_{t,k}^b$ to $W_{t,k}^b$.

THEOREM 3.2. *Let $\mathcal{M} := \{M_{i,j} : i\in[0:I], j\in[J]\}$ be a multi-set of matrices such that each $M_{i,j}$ or its transpose only takes value in $W_{:t}^b\bigcup G_{:t-1}^b\bigcup\mathcal{C}$ and $\deg\left(G_t^b;\mathcal{M}\right)=d$. Then there exist constants $\mu_1,\dots,\mu_S\in\mathbb{N}^+, S<\infty$ independent of b and a multi-set of matrices $\mathcal{Q}=\{Q_{s,i,j}, s\in[S], i\in[0:I], j\in[J]\}$ such that*

$$\operatorname{tr}\left(C\left(\bigotimes_{i\in[I]}\prod_{j\in[J]}M_{i,j}\right)\right)\prod_{j\in[J]}M_{0,j}=\sum_{s\in[S]}\mu_s\operatorname{tr}\left(C\left(\bigotimes_{i\in[0:I]}\prod_{j\in[J]}Q_{s,i,j}\right)\right)\prod_{j\in[J]}Q_{s,0,j},$$

where $C\in\mathcal{C}$ is a constant matrix, and $M_{s,i,j}\in W_{:t-1}^b\bigcup G_{:t-1}^b\bigcup\mathcal{C}$.

We present the complete version of these theorems and their proofs in Appendix A.1. The exact values of $I', J', c_{l,s}, C_{l,s}, L_s, \alpha_s, S, Q_{l,s,i,j}$ and $Q_{l,s,i}$ are also provided in the corresponding proofs.

In fact, these two theorems provide a recursive relationship for explicitly representing any quantity of the form

$$\operatorname{tr}\left(C\left(\bigotimes_{i\in[I]}\prod_{j\in[J]}M_{i,j}\right)\right)\prod_{j\in[J]}M_{0,j},\quad M_{i,j}\in W_{:t}^b\bigcup G_{:t}^b\bigcup\mathcal{C}\tag{3.3}$$

[2]The definition of $\mathcal{C}$ here is loose to keep the main body of the paper concise. We give a more detailed definition of $\mathcal{C}$ in Appendix A.1.

as the sum of many other terms of the same form

$$\operatorname{tr}\left(C\left(\bigotimes_{i\in[I]}\prod_{j\in[J]}M_{i,j}\right)\right)\prod_{j\in[J]}M_{0,j}=\sum_{s}\mu_s'\operatorname{tr}\left(C\left(\bigotimes_{i\in[0:I']}\prod_{j\in[J']}Q_{s,i,j}\right)\right)\prod_{j\in[J']}Q_{s,0,j},$$

where $Q_{s,i,j}\in W^b_{:t-1}\bigcup G^b_{:t-1}\bigcup\mathcal{C}$ and μ_s's' are some constants independent of b. Since $Q_{s,i,j}$ no longer takes value in $W^b_t\bigcup G^b_t$, we are able to reduce the time step by one. As a direct result, by recursively applying these two theorems, we are able to represent the expected value (conditioning on $\mathcal{F}_0$) of the term in (3.3) using learning rates, initial weights, ground-truth weights, and other constants matrices.

THEOREM 3.3. *Let $\mathcal{M}:=\{M_{i,j}: i\in[0:I], j\in[J]\}$ be a multi-set of matrices such that each $M_{i,j}$ or its transpose only takes value in $W^b_{:t}\bigcup G^b_{:t}\bigcup\mathcal{C}$. Then there exist constants $I',J',S,\overline{L}_s$ independent of b, $s\in[0:S]$ and a multi-set of matrices $\mathcal{Q}=\left\{Q_{l,s,i,j}, l\in[\overline{L}_s], s\in[S], i\in[0:I'], j\in[J'],\right\}$ such that*

$$\mathbb{E}\left[\operatorname{tr}\left(C\left(\bigotimes_{i\in[I]}\prod_{j\in[J]}M_{i,j}\right)\right)\prod_{j\in[J]}M_{0,j}\middle|\mathcal{F}_0\right]=\sum_{s\in[S]}Q_s\frac{1}{b^s}, \tag{3.4}$$

where $Q_s=\sum_{l\in[\overline{L}_s]}c_{l,s}\operatorname{tr}\left(C_{l,s}\left(\bigotimes_{i\in[I']}\prod_{j\in[J']}Q_{l,s,i,j}\right)\right)\prod_{j\in[J']}Q_{l,s,0,j}, s\in[0:S]$, $c_{l,s}$ is a constant, $C,C_{l,s}\in\mathcal{C}$ are constant matrices, and $Q_{l,s,i,j}\in W^b_0\bigcup\mathcal{C}$.

Again, the complete version of Theorem 3.3 and the exact values of these constants and matrices are presented in Appendix A.1.

3.1.2. Applications: Decreasing Property of the Variance of SG estimators. In this section, we use the theorems presented in Section 3.1.1 to show some applications of this framework. It is easy to verify that $\mathsf{var}\left(g^b_{t,k}\right)$, $\mathbb{E}\left[\mathcal{L}(w^b_t)\right]$ and $\mathsf{var}\left(\mathcal{L}(w^b_t)\right)$ can be written as the sum of several terms in the form of the left hand side of (3.4) by further taking expectation over the random initialization of weight matrices[3]. As a special case of Theorem 3.3, Theorem 3.4 shows that the variance of the SG estimators is a polynomial of $\frac{1}{b}$ without a constant term. This backs the important intuition that the variance is approximately inversely proportional to the mini-batch size b and provide much more precise relationship between the variance and the mini-batch size b.

THEOREM 3.4. *Given $t\in\mathbb{N}$, value $\mathsf{var}\left(g^b_{t,k}\right)$, $k\in[H]$ can be written as a polynomial of $\frac{1}{b}$ with degree at most $(D+1)^{(t+1)D}-1$ with no constant term. Formally, we have $\mathsf{var}\left(g^b_{t,k}\right)=\beta_1\frac{1}{b}+\cdots+\beta_r\frac{1}{b^r}$, where $r\leqslant 2(D+1)^{(t+1)D}-1$ and each β_i is a constant independent of b.*

One should note that the polynomial representation of $\mathsf{var}\left(g^b_{t,k}\right)$ does not have the constant term. This is intuitively correct since $\mathsf{var}\left(g^b_{t,k}\right)\to 0$ as $b\to\infty$. Therefore, to show that the variance is a decreasing function of b, we only need to show that the leading coefficient β_1 is non-negative. This is guaranteed by the fact that variance is always non-negative. We therefore have the next theorem.

[3]For example, for $i\in[H]$, we have

$$\mathsf{var}\left(g^b_{t,i}\right)=\mathbb{E}\left[\left\|g^b_{t,i}\right\|^2\right]-\left\|\mathbb{E}g^b_{t,i}\right\|^2=\mathbb{E}_{w_0}\left[\mathbb{E}\left[\operatorname{tr}\left(g^b_{t,i}\left(g^b_{t,i}\right)^T\right)\middle|\mathcal{F}_0\right]\right]-\left\|\mathbb{E}_{w_0}\left[\mathbb{E}\left[g^b_{t,i}\middle|\mathcal{F}_0\right]\right]\right\|^2.$$

THEOREM 3.5. *Given $t \in \mathbb{N}$, there exists a constant b_0 such that for all $b \geqslant b_0$, function* $\mathsf{var}\left(g_{t,k}^b\right), k \in [H]$ *is a decreasing function of b.*

The constant b_0 is the largest root of the equation $\beta_1 b^{r-1} + \beta_2 b^{r-2} + \cdots + \beta_r = 0$. See the proof of Theorem 3.5 in Appendix A.1 for more details. Although we cannot provide an explicit form of b_0, we can calculate it by the recursive relationship as provided in Theorems 3.1 and 3.2. We further numerically verify that b_0 is 1 in many setups (see Section 4 for more details). From the proofs we conclude that the scale of each β_i is of the order $\mathcal{O}(\|M\|)$, where M is a product of $W_{0,k}, W_k^*, k \in [H]$ and other constant matrices.

In conclusion, we provide a framework for recursively calculating the expected value of a general form that consists of SG estimators and weight matrices at time step t. As an application, we use our framework to represent the variance of SG estimators by a polynomial in $1/b$ and prove that the variance is a decreasing function of b when b is large. Readers should note that the framework here can handle $g_{t,k}^b$ and $W_{t,k}^b$ with any finite degree, and thus it provides much larger capability than just calculating the variance. As a result, similar to Theorems 3.4 and 3.5, we can show that the population loss $\mathcal{L}(w_t^b)$ at iteration t is also a polynomial in $1/b$ and is a decreasing function of b when b is large.

3.2. General Feed-forward Neural Networks. In this section, we discuss the extensions of our framework to feed-forward networks with general (non-polynomial) activation functions.

Note that for any smooth activation function σ^S (e.g., Sigmoid and Leaky ReLU), it is always possible to find a corresponding polynomial function, σ^P such that it approximates σ^S as closely as desired within a specified compact domain. This means that, regardless of the specific smooth activation function used, there exists a polynomially-activated function that can mimic its behavior within a certain range. This intuition leads to the following theorem.

THEOREM 3.6. *For any smooth activation function σ^S, $\epsilon > 0$ and time step $T \in \mathbb{N}^+$, there exists a polynomial σ^P (depending on ϵ, σ^S, and T) such that $\left\|g_{T,k}^S - g_{T,k}^P\right\| \leqslant \epsilon, k \in [H]$, where $g_{t,k}^S$ and $g_{t,k}^P$ are the stochastic gradient of the corresponding network's weight matrix in the k-th layer at time step t.*

The proof of the above theorem is deferred to Appendix A.1.4. Theorem 3.6 states that the SG estimators of a general neural network can be approximated arbitrarily well by the counterpart of a polynomially-activated function at any given time step T. This is a significant finding as it allows us to approximate the behavior of complex neural networks using simpler polynomial activation functions. Furthermore, when we combine this with the theorems presented in Section 3.1, which provide an exact representation of the SG estimators of any polynomially-activated function using only information available before training, we gain the ability to approximate the SG estimators of general networks arbitrarily well using only the known information at the initial time step $t = 0$.

This approximation has profound implications for our understanding of neural network behavior and offers potential avenues for designing more advanced optimization methods. See the discussions in Section 5 for more details.

4. Experiments. In this section, we present numerical results to support the theorems in Section 3, to backup the hypotheses discussed in the introduction, and provide further insights into the impact of the mini-batch size on the dynamics of

SGD. The experiments are conducted on four datasets and models that are relatively small due to the computational cost of using large models and datasets.

4.1. Datasets and Settings. For all experiments, we perform mini-batch SGD multiple times starting from the same initial weights and following the same choice of the learning rates and other hyper-parameters, if applicable. This enables us to calculate the variance of the gradient estimators and other statistics in each iteration, where the randomness comes only from different samples of SGD. The learning rate α_t is selected to be inversely proportional to iteration t, or fixed, depending on the task at hand.

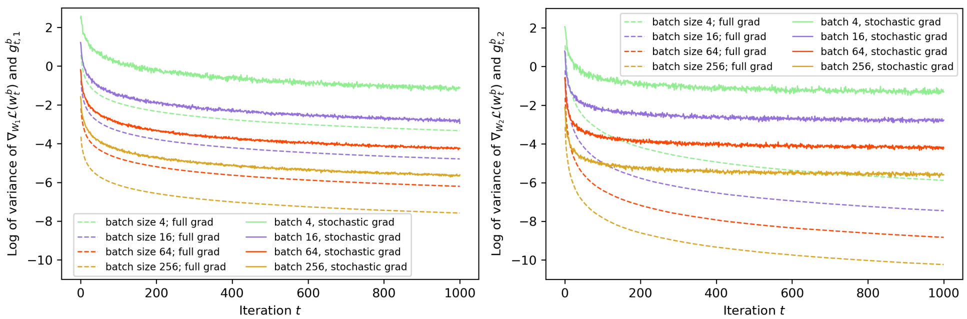
All models are implemented using PyTorch version 1.4 [32] and trained on NVIDIA 2080Ti/1080 GPUs. We have also tested several other random initial weights and ground-truth weights, and learning rates, and the results and conclusions are similar and not presented.

4.1.1. Synthetic Dataset. We build a synthetic dataset of standard normal samples to study the setting in Section 3.1. We fix the teacher network with 64 input neurons, 256 hidden neurons and 128 output neurons. We optimize the population L_2 loss by updating the two parameter matrices of the student network using online SGD, as stated in Section 3.1. In this case we have proved the functional form of the variance as a function of b and show the decreasing property of the variance of the SG estimators for large mini-batch sizes. However, we do not show the decreasing property for every b. With this experiment we confirm that the conjecture likely holds. In the experiment, we randomly select two initial weight matrices $W_{0,1}, W_{0,2}$ and the ground-truth weight matrices W_1^*, W_2^*. We run SGD for 1,000 iterations which appears to be a good number for convergence while there are 1,000 runs of SGD in total to again give a p-value below 0.05. We record all statistics at every iteration. The learning rate is chosen to be $\alpha_t = \frac{1}{10t}, t \in [1000]$ for the same reason as in the regression experiment.

4.1.2. MNIST Dataset. The MNIST dataset is to recognize digits in handwritten images of digits. We use all 60,000 training samples and 10,000 validation samples of MNIST. The images are normalized by mapping each entry to $[-1, 1]$. We build a three-layer fully connected neural network with 1024, 512 and 10 neurons in each layer. For the two hidden layers, we use the ReLU activation function. The last layer is the softmax layer which gives the prediction probabilities for the 10 digits. We use mini-batch SGD to optimize the cross-entropy loss of the model. The model deviates from our analytical setting since it has non-linear activations, it has the cross-entropy loss function (instead of L_2), and empirical loss (as opposed to population). MNIST is selected due to its fast training and popularity in deep learning experiments. The goal is to verify the results in this different setting and to back up our hypotheses.

We run SGD for 1,000 epochs on the training set which is enough for convergence. The learning rate is a constant set to $3 \cdot 10^{-3}$ (which has been tuned). For the experiment in Figure 3, there are in total 100 runs to give us the p-value below 0.05. For the experiment in Figure 2(a), we randomly select five different initial points and we have 50 runs for each initial point. For the experiment corresponding to Figure 2(b), we choose $\alpha = 8$ and $\sigma = 2$ as in [37]. The initial weights and other hyper-parameters are chosen to be the same as in Figure 3.

4.1.3. Yelp Review Dataset. The Yelp Review dataset from the Yelp Dataset Challenge [42] contains 1,569,264 samples of customer reviews with positive/negative sentiment labels. We use 10,000 samples as our training set and 1,000 samples as the



(a) Variance of gradients with respect to W_1 (b) Variance of gradients with respect to W_2

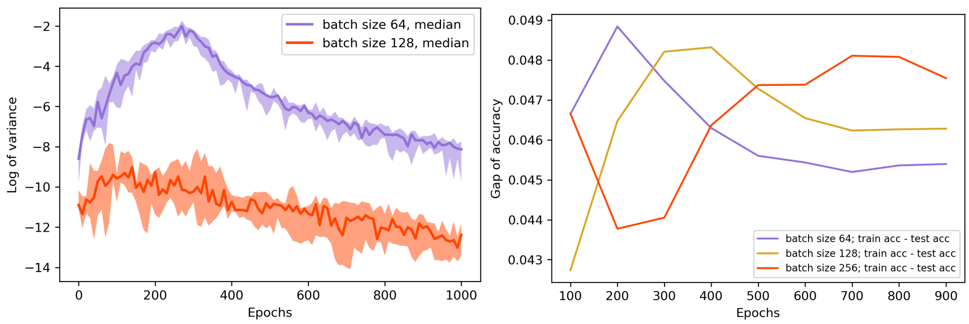
Fig. 1: Experimental results for the Synthetic dataset. **Left:** $\log\left(\mathsf{var}\left(g_{t,1}^b|\mathcal{F}_0\right)\right)$ and $\log\left(\mathsf{var}\left(\nabla_{W_1}\mathcal{L}(W_{t,1}^b, W_{t,2}^b)\,|\mathcal{F}_0\right)\right)$ vs iteration t. **Right:** $\log\left(\mathsf{var}\left(g_{t,2}^b|\mathcal{F}_0\right)\right)$ and $\log\left(\mathsf{var}\left(\nabla_{W_2}\mathcal{L}(W_{t,1}^b, W_{t,2}^b)\,|\mathcal{F}_0\right)\right)$ vs iteration t.

validation set. We use XLNet [41] to perform sentiment classification on this dataset. Our XLNet has 6 layers, the hidden size of 384, and 12 attention heads. There are in total 35,493,122 parameters. We intentionally reduce the number of layers and hidden size of XLNet and select a relatively small size of the training and validation sets since training of XLNet is very time-consuming ([41] train on 512 TPU v3 chips for 5.5 days) and we need to train the model for multiple runs. This setting allows us to train our model in several hours on a single GPU card. We train the model using the Adam weight decay optimizer, and some other techniques, as suggested in Table 8 of [41]. This dataset represents sequential data where we further consider the hypotheses.

We randomly select a set of initial parameters and run Adam with two different mini-batch sizes of 32 and 64. For computational tractability reasons, for each mini-batch size there are in total of 100 runs and each run corresponds to 20 epochs. We record the variance of the stochastic gradient, loss and accuracy in every step of Adam. The statistics reported in Figure 4 are averaged through each epoch. In all experiments, the learning rate is set to be $4 \cdot 10^{-5}$ and the ϵ parameter of Adam is set to be 10^{-8} (these two have been tuned). The stochastic gradients of all parameter matrices are clipped with threshold 1 in each iteration. We use the same setup for the learning rate warm-up strategy as suggested in [41]. The maximum sequence length is set to be 128 and we pad the sequences with length smaller than 128 with zeros.

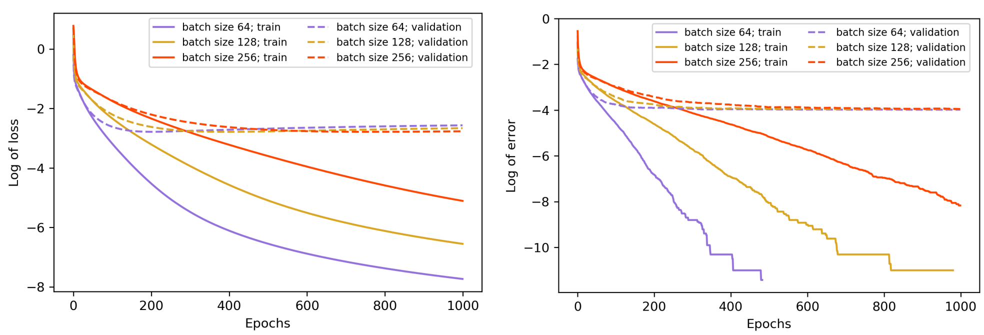
4.2. Discussion. Under the two-layer linear network setting with the synthetic dataset, Figure 1 verifies that the variance of the SG estimators and full gradients are all strictly decreasing functions of b for all iterations. This figure also empirically shows that the constant b_0 in Theorem 3.5 could be as small as $b_0 = 4$. In fact, we also experiment with the mini-batch size of 1 and 2, and the decreasing property remains to hold. We also test this on multiple choices of initial weights and learning rates and this pattern remains clear.

In aforementioned two experiments we use SGD in its original form by randomly sampling mini-batches. In deep learning with large-scale training data such a strategy is computationally prohibitive and thus samples are scanned in a cyclic order which implies fixed mini-batches are processed many times. Therefore, in the next two datasets we perform standard "epoch" based training to empirically study the remaining



(a) Different initial weights (b) Gap of accuracy (zoomed-in)

Fig. 2: Experimental results for the MNIST dataset. **Left:** The median, min, and max of the log of variance of the SG estimators for two different mini-batch sizes (distinguished by colors) and five different initial weights. The solid lines show the median of all five initial weights while the highlighted regions show the min and max of the log of variance. **Right:** The gap of accuracy on training and test sets vs epochs starting from epoch 100.



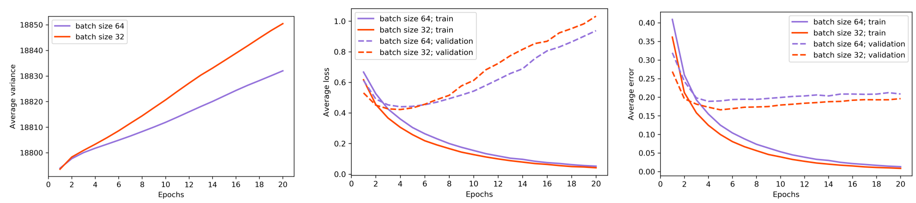
(a) Log of loss for training and validation sets (b) Log of error for training and validation sets

Fig. 3: Experimental results for the MNIST dataset. **Left:** The log of the training and validation loss vs epochs. **Right:** The log of training and validation error vs epochs. Here error is defined as one minus predicting accuracy. The plot does not show the epochs if error equals to zero.

two hypotheses discussed in the introduction (decreasing loss and error as a function of b) and sensitivity with respect to the initial weights. Note that we are using cross-entropy loss in the MNIST dataset and the Adam optimizer in the Yelp dataset and thus these experiments do not meet all of the assumptions of the analysis in Section 3.

As shown in Figure 2(a), we run SGD with two batch sizes 64 and 128 on five different initial weights. This plot shows that, even the smallest value of the variance among the five different initial weights with a mini-batch size of 64, is still larger than the largest variance of mini-batch size 128. We observe that the sensitivity to the initial weights is not large. This plot also empirically verifies our conjecture in the introduction that the variance of the SG estimators is a decreasing function of the mini-batch size, for all iterations of SGD in a general deep learning model.

In addition, we also conjecture that there exists the decreasing property for the



(a) Variance of SG (b) Training/validation loss (c) Training - validation error

Fig. 4: Experimental results for the XLNet model on the Yelp dataset. **Left:** The variance of SG estimators vs epochs. **Middle:** The training and validation loss vs epochs. **Right:** The training and validation error vs epochs.

expected loss, error and the generalization ability with respect to the mini-batch size. Figure 3(a) shows that the expected loss (again, randomness comes from different runs of SGD through the different mini-batches with the same initial weights and learning rates) on the training set is a decreasing function of b. However, this decreasing property does not hold on the validation set when the loss tends to be stable or increasing, in other words, the model starts to be over-fitting. We hypothesize that this is because the learned weights start to bounce around a local minimum when the model is over-fitting. As the larger mini-batch size brings smaller variance, the weights are closer to the local minimum found by SGD, and therefore yield a smaller loss function value. Figure 3(b) shows that both the expected error on training and validation sets are decreasing functions of b.

Figure 2(b) exhibits a relationship between the model's generalization ability and the mini-batch size. As suggested by [37], we build a test set by distorting the 10,000 images of the validation set. The prediction accuracy is obtained on both training and test sets and we calculate the gap between these two accuracies every 100 epochs. We use this gap to measure the model generalization ability (the smaller the better). Figure 2(b) shows that the gap is an increasing function of b starting at epoch 500, which partially aligns with our conjecture regarding the relationship between the generalization ability and the mini-batch size. We also test this on multiple choices of the hyper-parameters which control the degree of distortion in the test set and this pattern remains clear.

Figure 4 shows the similar phenomenon that the variance of stochastic estimators and the expected loss and error on both training and validation sets are decreasing functions of b even if we train XLNet using Adam. This example gives us confidence that the decreasing properties are not merely restricted on shallow neural networks or vanilla SGD algorithms. They actually appear in many advanced models and optimization methods.

5. Discussion and Future Work. We study the dynamics of SGD by explicitly representing the important quantities of SGD using the mini-batch size and initial weights. For multi-layer polynomially-activated network, we are able to build frameworks that recursively calculate general forms of the product of the weight matrices and SG estimators between consecutive iterations. We extend polynomial activation function to general activation functions with mil assumptions and further theoretically prove that the variance conjecture holds. Experiments are performed on multiple models and datasets to verify our claims and their applicability to practical settings.

Besides, we also empirically address the conjectures about the expected loss and the generalization ability.

We provide mathematical tools to calculate and represent the product of the stochastic gradients estimators and weight matrices in the t-th step (and not a single step), which is non-trivial and requires a sophisticated mathematical proof. These tools can be extended to calculate any form that has a polynomial relationship to the model parameters w_t^b, e.g. expectation/variance of the loss function, norm of the SG estimator to any finite degree. We can also derive other properties of the dynamics of SGD by using these tools.

One possible application of the results is to help tighten the convergence rates of SGD and develop better variance reduction methods. Current analyses of SGD convergence rely on two constants M and M_V such that $\mathsf{var}\left(g_t^b\right) \leqslant M + M_V \left\|\nabla L(w_t^b)\right\|^2$. But it is unclear what are the exact values of M and M_V (see Assumption 4.3 of [5] and the context therein). It is a common practice to take relatively large M and M_V to make sure the above bound holds. However, this leads to a relatively poor convergence rate of the SGD algorithm. Our frameworks are able to explicitly calculate $\mathsf{var}\left(g_t^b\right)$ and $\left\|\nabla L(w_t^b)\right\|^2$ by recursive formulas and thus to provide optimal values for M and M_V.

Another challenging research direction is to theoretically and explicitly investigate the generalization ability during training of SGD. There are existing works studying the relationship between the variance of the stochastic gradients and the generalization ability [10, 29]. Together with the frameworks developed herein, it would be possible to tighten the generalization bounds of a neural network by explicit variance and other quantities. We can further choose an optimal mini-batch size which minimizes the generalization ability by solving a polynomial equation if we have a more precise relationship between the variance and the generalization ability.

Further interesting work is to extend our techniques to more complicated and sophisticated networks as we discuss in Section 3.2. Although the underlying model of this paper corresponds to deep polynomially-activated networks in a strict manner and to general neural networks in an approximate sense, we are able to show a deeper relationship between the variance and the mini-batch size, the polynomial in $1/b$, while the common knowledge is simply that the variance is proportional to $1/b$. The extension to other optimization algorithms, like Adam and Gradient Boosting Machines, are also very attractive. We hope our theoretical framework can serve as a tool for future research of this kind.

Appendix A. Lemmas and Proofs.

A.1. Proofs for Results in 3.1. We provide an outlined proof of the two-layer linear network in Appendix A.1.1. Due to the page limit, we only present the statement of some lemmas and theorems here. Please refer to the Appendix of [2] for full proofs. We defer the extension from linear networks to polynomially-activated networks in Appendix A.1.2.

A.1.1. Two-layer Linear Networks. Given a distribution $\mathcal{D}$ in $\mathbb{R}^p$, we consider the population loss $\mathcal{L}(w) = \mathbb{E}_{x\sim\mathcal{D}}\left[\frac{1}{2}\left\|W_2W_1x - W_2^*W_1^*x\right\|^2\right]$ under the teacher-student learning framework [14] with $w = (W_1, W_2)$ a tuple of two matrices. Here $W_1 \in \mathbb{R}^{p_1\times p}$ and $W_2 \in \mathbb{R}^{p_2\times p_1}$ are parameter matrices of the student network and W_1^* and W_2^* are the fixed ground-truth parameters of the teacher network. We use online SGD to minimize the population loss $\mathcal{L}(w)$. Formally, we first choose a mini-batch size b and initial weight matrices $\{W_{0,1}, W_{0,2}\}$; in each iteration t, we independently draw a mini-batch $\mathcal{B}_t^b := \left\{x_{t,i}^b : i \in [b]\right\}$ of b samples from $\mathcal{D}$ and update the weight matrices by $W_{t+1,1}^b = W_{t,1}^b - \alpha_t g_{t,1}^b$ and $W_{t+1,2}^b = W_{t,2}^b - \alpha_t g_{t,2}^b$, where

$$g_{t,1}^b := \frac{1}{b}\sum_{i=1}^{b} \nabla_{W_{t,1}^b}\left(\frac{1}{2}\left\|W_{t,2}^b W_{t,1}^b x_{t,i}^b - W_2^* W_1^* x_{t,i}^b\right\|^2\right) = \frac{1}{b}\sum_{i=1}^{b}\left(W_{t,2}^b\right)^T \mathcal{W}_t^b x_{t,i}^b \left(x_{t,i}^b\right)^T, \tag{A.1}$$

$$g_{t,2}^b := \frac{1}{b}\sum_{i=1}^{b} \nabla_{W_{t,2}^b}\left(\frac{1}{2}\left\|W_{t,2}^b W_{t,1}^b x_{t,i}^b - W_2^* W_1^* x_{t,i}^b\right\|^2\right) = \frac{1}{b}\sum_{i=1}^{b}\mathcal{W}_t^b x_{t,i}^b \left(x_{t,i}^b\right)^T \left(W_{t,1}^b\right)^T. \tag{A.2}$$

Here $\mathcal{W}_t^b := W_{t,2}^b W_{t,1}^b - W_2^* W_1^*$ denotes the gap between the product of model weights and ground-truth weights and the derivation follows from the formulas in [33].

To recap, we use $\deg(A;\mathcal{M})$ to denote the number of appearances of matrix A and its transpose A^T in a multi-set of matrices $\mathcal{M} = \{M_1, \ldots, M_n\}$. Mathematically, we have $\deg(A;\mathcal{M}) := \sum_{i\in[n]}\left(\mathbb{I}\{A = A_i\} + \mathbb{I}\left\{A^T = A_i\right\}\right)$. We further denote $\deg(\mathcal{A};\mathcal{M}) := \sum_{A\in\mathcal{A}} \deg(A;\mathcal{M})$ for any set of matrices $\mathcal{A}$. We denote $W_t^b := \left\{W_{t,1}^b, W_{t,2}^b\right\}$, $W^* := \{W_1^*, W_2^*\}$ and $G_t^b := \left\{g_{t,1}^b, g_{t,2}^b\right\}$.

In Section 3.1, we use $\mathcal{C}$ to denote the infinite set of all non-random matrices given $\mathcal{F}_0$. Here we provide the precise definition of $\mathcal{C}$ as follows. For $n \in \mathbb{N}^+$, we use $e_{n,i}, i \in [n]$ to denote the i-th unit vector of $\mathbb{R}^n$. We denote $\mathcal{I} = \{I_n : n \in \mathbb{N}^+\}$ as the collection of identity matrices and we define a set of (infinite many) matrices

$$\mathcal{C} := \left\{\begin{array}{l} \mathbb{E}_{x_{t,i}^b\sim\mathcal{D}, i\in[b]}\left[(e_{p,u}^T z_0)(e_{p,v}^T \overline{z}_0)\left[\left(y_1\overline{y}_1^T\right)\otimes\cdots\otimes\left(y_m\overline{y}_m^T\right)\otimes\left(z_1\overline{z}_1^T\right)\otimes\cdots\left(z_n\overline{z}_n^T\right)\right]\right] : \\ \qquad y_i = e_{p,j_1^i}\otimes\cdots\otimes e_{p,j_{m_i}^i}\otimes x_{t,s_i}^b\otimes e_{p,k_1^i}\otimes\cdots\otimes e_{p,k_{n_i}^i}, \\ \qquad \overline{y}_i = e_{p,\overline{j}_1^i}\otimes\cdots\otimes e_{p,\overline{j}_{m_i}^i}\otimes x_{t,\overline{s}_i}^b\otimes e_{p,\overline{k}_1^i}\otimes\cdots\otimes e_{p,\overline{k}_{n_i}^i}, \\ \qquad z_0 \in \left\{x_{t,i}^b : i\in[b]\right\}\bigcup\{e_{p,u}\}, \overline{z}_0 \in \left\{x_{t,i}^b : i\in[b]\right\}\bigcup\{e_{p,v}\}, u,v\in[p], \\ \qquad z_j, \overline{z}_j \in \left\{x_{t,i}^b : i\in[b]\right\}, j\in[n], \\ \qquad j_\alpha^i, \overline{j}_\alpha^i, k_\beta^i, \overline{k}_\beta^i \in [p], \alpha\in[m_i], \beta\in[n_i], i\in[m], \\ \qquad m_i, n_i \in \mathbb{N}, s_i, \overline{s}_i \in [b], i\in[m], \\ \qquad m, n\in\mathbb{N}, t\in\mathbb{N}^+ \end{array}\right\}$$

where p is the dimension of the samples and $x_{t,s}^b, s \in [b]$ are the random samples we use to build the stochastic gradient at step t and thus every element of $\mathcal{C}$ is a constant matrix under $\mathcal{F}_0$. Note that $\mathcal{C}$ is a union over all $m, n, m_i, n_i \in \mathbb{N}$ and $t \in \mathbb{N}^+$. We also point out that when $z_0 = e_{p,u}, \overline{z}_0 = e_{p,v}$, the leading scalar terms are 1. We also denote $\mathcal{E} := \left\{e_{p,i}e_{p,j}^T : i,j \in [p]\right\}$ and $\overline{\mathcal{C}} := \mathcal{C}\bigcup\mathcal{I}\bigcup\mathcal{E}$. Note that every element of $\overline{\mathcal{C}}$ is a non-random matrix under $\mathcal{F}_0$ and $\overline{\mathcal{C}}$ is an infinite set of matrices that we use in the following proofs as auxiliary matrices.

Let $g_{t,1,s}^b := \left(W_{t,2}^b\right)^T \cdot \mathcal{W}_t^b \cdot \left(x_{t,s}^b\left(x_{t,s}^b\right)^T\right)$ and $g_{t,2,s}^b := \mathcal{W}_t^b \cdot \left(x_{t,s}^b\left(x_{t,s}^b\right)^T\right)\cdot W_{t,1}^b, s \in [b]$ denote the stochastic gradient with respect to the sample $x_{t,s}^b$ at time step t. We have $g_{t,i}^b = \frac{1}{b}\sum_{s\in[d]} g_{t,i,s}^b, i = 1,2$. Recall that we denote $W_t^b = \left\{W_{t,1}^b, W_{t,2}^b\right\}$, $W^* = \{W_1^*, W_2^*\}$ and $G_t^b = \left\{g_{t,1}^b, g_{t,2}^b\right\}$ in Section 3.1. We further denote $\overline{G}_t^b = \left\{g_{t,i,s}^b : s\in[b], i = 1,2\right\}$ and $X_t^b = \left\{x_{t,s}^b\left(x_{t,s}^b\right)^T : s \in [b]\right\}$. For simplicity, we denote $G_{t_1:t_2}^b := \bigcup_{t=t_1}^{t_2} G_t^b$ and $W_{t_1:t_2}^b := \bigcup_{t=t_1}^{t_2} W_t^b$.

Throughout the discussion of this section, we define the term that a matrix A "takes values in" or "belongs to" a multi-set $\mathcal{A}$ if either A or A^T are in $\mathcal{A}$. We also abuse the notation $A \in \mathcal{A}$ to denote A is in $\mathcal{A}$ or A^T is in A.

LEMMA A.1. *For matrices $M_{i,j}, i \in [m], j \in [n]$ with appropriate dimensions, we have* $\bigotimes_{i\in[m]}\left(\prod_{j\in[n]} M_{i,j}\right) = \prod_{j\in[n]}\left(\bigotimes_{i\in[m]} M_{i,j}\right)$.

Remark. If we view the multi-set $\mathcal{M} := \{M_{i,j}, i \in [m], j \in [n]\}$ as a matrix of matrices

$$\mathcal{M}: \begin{bmatrix} M_{1,1} & M_{1,2} & M_{1,3} & \cdots & M_{1,n} \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ M_{m,1} & M_{m,2} & M_{m,3} & \cdots & M_{m,n} \end{bmatrix},$$

then $\bigotimes_{i\in[m]}\left(\prod_{j\in[n]} M_{i,j}\right)$ can be regarded as first multiplying the entries of $\mathcal{M}$ within each row and then using the Kronecker product to multiply all of the rows. Similarly, $\prod_{j\in[n]}\left(\bigotimes_{i\in[m]} M_{i,j}\right)$ can be regarded as first using the Kronecker product to multiply all the entries of a column, then multiplying all the rows. Lemma A.1 shows that these two calculations on multi-set $\mathcal{M}$ give the same resulting matrices. We frequently use this lemma in the following proofs. We give illustrations of the multi-sets to help readers better understand and follow the proofs.

LEMMA A.2. *Given two distributions $\mathcal{D}_1$ and $\mathcal{D}_2$ in $\mathbb{R}^{p_1}$ and $\mathbb{R}^{p_2}$, respectively. Given $y_1, \ldots, y_m \sim \mathcal{D}_1$, $z_1, \ldots z_n \sim \mathcal{D}_2$ and constant matrices $D_0, \ldots, D_n, A_1, \ldots, A_m$ with appropriate dimensions, we have*

$$\begin{aligned}&\mathbb{E}_{y_i\sim\mathcal{D}_1, z_j\sim\mathcal{D}_2}\left[D_0 z_1 z_n^T D_n \left(z_1^T D_1 z_2\right)\cdots\left(z_{n-1}^T D_{n-1} z_n\right)\left(y_m^T A_m y_1\right)\left(y_1^T A_1 y_2\right)\cdots\left(y_{m-1}^T A_{m-1} y_m\right)\right] \\ &= \sum_{u\in[p_1], v\in[p_2]}\left[D_0 e_{p_1,u} e_{p_2,v}^T D_n \operatorname{tr}\left(C_{u,v}\left(\left(\bigotimes_{i=0}^{m-1} A_i\right)\otimes\left(\bigotimes_{j=1}^{n-1} D_i\right)\right)\right)\right]\end{aligned}$$

for some constant matrices $C_{u,v}$ specified in the proof.

LEMMA A.3. *Let $\mathcal{M} := \{M_{i,j} : i \in [0:m], j \in [n]\}$ be a multi-set of matrices such that each $M_{i,j}$ or its transpose only takes value in $W_{0:t}^b \bigcup \overline{G}_t^b \bigcup G_{0:(t-1)}^b \bigcup W^* \bigcup \overline{\mathcal{C}}$ and $\deg\left(\overline{G}_t^b; \mathcal{M}\right) = d$ (here d, m, n are constants independent of b). Then for*

$$m' := m + d - 2, \quad n' := 6mn(d+1), \quad L := 2^d p^{d'(m-1)+2},$$

where $d' = \deg\left(\overline{G}_t^b; \{M_{i,j} : i \in [m], j \in [n]\}\right)$, there exist multi-sets of matrices

$$\mathcal{Q}_l := \{Q_{l,u,v} : u \in [0:m'], v \in [n']\}, l \in [L]$$

such that

$$\mathbb{E}\left[\operatorname{tr}\left(C\left(\bigotimes_{i\in[m]}\left(\prod_{j\in[n]} M_{i,j}\right)\right)\right)\prod_{j\in[n]} M_{0,j}\middle|\mathcal{F}_t^b\right] = \sum_{l\in[L]} c_l \operatorname{tr}\left(C_l\left(\bigotimes_{u\in[m']}\left(\prod_{v\in[n']} Q_{l,u,v}\right)\right)\right)\prod_{v\in[n']} Q_{l,0,v},$$

where $c_l \in \{-1, +1\}$, $C, C_l \in \mathcal{C}$ and $Q_{l,u,v}$ only takes value in $W_{0:t}^b \bigcup G_{0:(t-1)}^b \bigcup W^* \bigcup \overline{\mathcal{C}}$, $u \in [0:m'], v \in [n'], l \in [L]$. Further, for each $l \in [L]$ we have

$$\begin{aligned}\deg\left(\overline{G}_t^b; \mathcal{Q}_l\right) &= 0, \\ \deg\left(W_t^b; \mathcal{Q}_l\right) &\leqslant \deg\left(W_t^b; \mathcal{M}\right) + 3d, \\ \deg\left(W^*; \mathcal{Q}_l\right) &\leqslant \deg\left(W^*; \mathcal{M}\right) + 2d, \\ \deg\left(W_t^b; \mathcal{Q}_l\right) + \deg\left(W^*; \mathcal{Q}_l\right) &= \deg\left(W_t^b; \mathcal{M}\right) + \deg\left(W^*; \mathcal{M}\right) + 3d, \\ \deg\left(W_f^b; \mathcal{Q}_l\right) &= \deg\left(W_f^b; \mathcal{M}\right), \quad f \in [0:t-1], \\ \deg\left(G_f^b; \mathcal{Q}_l\right) &= \deg\left(G_f^b; \mathcal{M}\right), \quad f \in [0:t-1].\end{aligned}$$

THEOREM A.4 (complete version of two-layer linear networks for Theorem 3.1). *Let* $\mathcal{M} := \{M_{i,j} : i \in [0:m], j \in [n]\}$ *be a multi-set of matrices such that each* $M_{i,j}$ *or its transpose only takes value in* $W^b_{0:t} \bigcup G^b_{0:t} \bigcup W^* \bigcup \overline{\mathcal{C}}$ *and* $\deg\left(G^b_t; \mathcal{M}\right) = d$ *(here* d, m, n *are constants independent of* b*). Then for* $m' := m+d-2$ *and* $n' := 6mn(d+1)$*, there exist a constant* L *independent of* b *and multi-sets of matrices*

$$\mathcal{Q}_{l,s} := \left\{Q_{l,s,u,v} : u \in [0:m'], v \in [n']\right\}, l \in [L], s \in [0:d]$$

such that

$$\mathbb{E}\left[\operatorname{tr}\left(C\left(\bigotimes_{i\in[m]}\left(\prod_{j\in[n]} M_{i,j}\right)\right)\right)\prod_{j\in[n]} M_{0,j}\middle|\mathcal{F}^b_t\right] = \tilde{\alpha}_0 + \tilde{\alpha}_1\frac{1}{b} + \cdots + \tilde{\alpha}_d\frac{1}{b^d},$$

where $\widetilde{\alpha}_s = \sum_{l\in[L]} c_{l,s} \operatorname{tr}\left(C_{l,s}\left(\bigotimes_{u\in[m']}\left(\prod_{v\in[n']} Q_{l,s,u,v}\right)\right)\right)\prod_{v\in[n']} Q_{l,s,0,v}$*,* $s \in [0:d]$*,* $c_{l,s}$ *is a constant,* $C_{l,s} \in \mathcal{C}$ *and* $Q_{l,s,u,v}$ *only takes value in* $W^b_{0:t} \bigcup G^b_{0:(t-1)} \bigcup W^* \bigcup \overline{\mathcal{C}}$*. Further, we have*

$$\begin{aligned}
\deg\left(G^b_t; \mathcal{Q}_{l,s}\right) &= 0,\\
\deg\left(W^b_t; \mathcal{Q}_{l,s}\right) &\leqslant \deg\left(W^b_t; \mathcal{M}\right) + 3d,\\
\deg\left(W^*; \mathcal{Q}_{l,s}\right) &\leqslant \deg\left(W^*; \mathcal{M}\right) + 2d,\\
\deg\left(W^b_t; \mathcal{Q}_{l,s}\right) + \deg\left(W^*; \mathcal{Q}_{l,s}\right) &= \deg\left(W^b_t; \mathcal{M}\right) + \deg\left(W^*; \mathcal{M}\right) + 3d,\\
\deg\left(W^b_f; \mathcal{Q}_{l,s}\right) &= \deg\left(W^b_f; \mathcal{M}\right), \quad f \in [0, t-1],\\
\deg\left(G^b_f; \mathcal{Q}_{l,s}\right) &= \deg\left(G^b_f; \mathcal{M}\right), \quad f \in [0, t-1],\\
\deg\left(W^*; \mathcal{Q}_{l,s}\right) &= \deg\left(W^*; \mathcal{M}\right).
\end{aligned}$$

THEOREM A.5 (complete version of two-layer linear networks for Theorem 3.2). *Let* $\mathcal{M} := \{M_{i,j} : i \in [0:m], j \in [n]\}$ *be a multi-set of matrices such that each* $M_{i,j}$ *or its transpose only takes value in* $W^b_{0:t} \bigcup G^b_{0:(t-1)} \bigcup W^* \bigcup \overline{\mathcal{C}}$ *and* $\deg\left(W^b_t; \mathcal{M}\right) = d$ *(here* d, m, n *are constants independent of* b*) and* $C \in \mathcal{C}$*. Then there exist multi-sets of matrices* $\mathcal{M}_k := \{M_{k,i,j} : i \in [0:m], j \in [n]\}, k \in [2^d]$ *such that*

$$\operatorname{tr}\left(C\left(\bigotimes_{i\in[m]}\left(\prod_{j\in[n]} M_{i,j}\right)\right)\right)\prod_{j\in[n]} M_{0,j} = \sum_{k\in[2^d]} \overline{\alpha}_k \operatorname{tr}\left(C\left(\bigotimes_{i\in[m]}\left(\prod_{j\in[n]} M_{k,i,j}\right)\right)\right)\prod_{j\in[n]} M_{k,0,j},$$

where $\overline{\alpha}_k, k \in [2^d]$ *are constants and each* $M_{k,i,j}$ *only takes value in*

$$W^b_{0:(t-1)} \bigcup G^b_{0:(t-1)} \bigcup W^* \bigcup \overline{\mathcal{C}}.$$

Further, for each $k \in [2^d]$ *we have*

$$\begin{aligned}
\deg\left(G^b_{t-1}; \mathcal{M}_k\right) &\leqslant \deg\left(G^b_{t-1}; \mathcal{M}\right) + d,\\
\deg\left(W^b_{t-1}; \mathcal{M}_k\right) &\leqslant \deg\left(W^b_{t-1}; \mathcal{M}\right) + d,\\
\deg\left(G^b_{t-1}; \mathcal{M}_k\right) + \deg\left(W^b_{t-1}; \mathcal{M}_k\right) &= \deg\left(G^b_{t-1}; \mathcal{M}\right) + \deg\left(W^b_{t-1}; \mathcal{M}\right) + d,\\
\deg\left(G^b_f; \mathcal{M}_k\right) &= \deg\left(G^b_f; \mathcal{M}\right), \quad f \in [0:(t-2)],\\
\deg\left(W^b_f; \mathcal{M}_k\right) &= \deg\left(W^b_f; \mathcal{M}\right), \quad f \in [0:(t-2)],\\
\deg\left(W^*; \mathcal{M}_k\right) &= \deg\left(W^*; \mathcal{M}\right).
\end{aligned}$$

Proof. We simply use the fact that $W_{t,i}^b = W_{t-1,i}^b - \alpha_t g_{t-1,i}^b, i = 1, 2$. Note that $\deg\left(W_t^b; \mathcal{M}\right) = d$, by replacing all appearance of $W_{t,i}^b$ in

$$\operatorname{tr}\left(C\left(\bigotimes_{i\in[m]}\left(\prod_{j\in[n]} M_{i,j}\right)\right)\right)\prod_{j\in[n]} M_{0,j}$$

with $\left(W_{t-1,i}^b - \alpha_t g_{t-1,i}^b\right)$ and expand all the parentheses, we get 2^d terms in the form of

$$\operatorname{tr}\left(C\left(\bigotimes_{i\in[m]}\left(\prod_{j\in[n]} M_{k,i,j}\right)\right)\right)\prod_{j\in[n]} M_{0,j}.$$

The constant $\overline{\alpha}_k$ comes from the multiplication of α_t's. ∎

THEOREM A.6 (complete version of two-layer linear networks for Theorem 3.3). *Let $\mathcal{M}^t := \left\{M_{i,j}^t : i \in [0:m_t], j \in [n_t]\right\}$ be a multi-set of matrices such that each $M_{i,j}^t$ or its transpose only takes value in $W_{0:t}^b \bigcup G_{0:t}^b \bigcup W^* \bigcup \overline{\mathcal{C}}$ (here m_t, n_t are constants independent of b) and $C_t \in \mathcal{C}$. Then there exist constants $q_t, m_t', n_t', L_{t,s}, s \in [0:q_t]$ that are independent of b and multi-sets of matrices*

$$\mathcal{M}_{l,s}^t := \left\{M_{l,s,u,v}^t : u \in [0:m_t'], v \in [n_t']\right\}, s \in [q_t]$$

such that

$$\mathbb{E}\left[\operatorname{tr}\left(C_t\left(\bigotimes_{i\in[m_t]}\left(\prod_{j\in[n_t]} M_{i,j}^t\right)\right)\right)\prod_{j\in[n_t]} M_{0,j}^t\middle|\mathcal{F}_0\right] = \alpha_{t,0} + \alpha_{t,1}\frac{1}{b} + \cdots + \alpha_{t,q_t}\frac{1}{b^{q_t}},$$

where $\alpha_{t,s} = \sum_{l\in[L_{t,s}]} c_{t,l,s}\operatorname{tr}\left(C_{t,l,s}\left(\bigotimes_{u\in[m_t']}\left(\prod_{v\in[n_t']} M_{l,s,u,v}^t\right)\right)\right)\prod_{v\in[n_t']} M_{l,s,0,v}^t$, $s \in [0:q_t]$, $c_{t,l,s}$ is a constant, $C_{t,l,s} \in \mathcal{C}$ and $M_{l,s,u,v}^t$ only takes value in $W_0^b \bigcup W^* \bigcup \overline{\mathcal{C}}$. Further, we have

$$q_t \leqslant \sum_{f\in[0:t]}\left(\frac{3^{f+1}-1}{2}\deg\left(G_f^b; \mathcal{M}^t\right) + \frac{3^f - 1}{2}\deg\left(W_f^b; \mathcal{M}^t\right)\right).$$

THEOREM A.7 (Two-layer linear network version for Theorem 3.4). *Given $t \in \mathbb{N}$, value $\mathsf{var}\left(g_{t,i}^b\right), i = 1, 2$ can be written as a polynomial of $\frac{1}{b}$ with degree at most $3^{t+1} - 1$ with no constant term. Formally, we have $\mathsf{var}\left(g_{t,i}^b\right) = \beta_1\frac{1}{b} + \cdots + \beta_r\frac{1}{b^r}$, where $r \leqslant 3^{t+1} - 1$ and each β_i is a constant independent of b.*

Proof. We only show the case for $g_{t,1}^b$ since the proof for $g_{t,2}$ can be tackled similarly. Note that

$$\begin{aligned}\mathsf{var}\left(g_{t,1}^b\right) &= \mathbb{E}\left\|g_{t,1}^b\right\|^2 - \left\|\mathbb{E}\left[g_{t,1}^b\right]\right\|^2 = \mathbb{E}\left[\mathbb{E}\left[\left\|g_{t,1}^b\right\|^2\middle|\mathcal{F}_0\right]\right] - \left\|\mathbb{E}\left[\mathbb{E}\left[g_{t,1}^b\middle|\mathcal{F}_0\right]\right]\right\|^2\\ &= \mathbb{E}\left[\mathbb{E}\left[\operatorname{tr}\left(\left(g_{t,1}^b\right)^T g_{t,1}^b\right)\middle|\mathcal{F}_0\right]\right] - \left\|\mathbb{E}\left[\mathbb{E}\left[g_{t,1}^b\middle|\mathcal{F}_0\right]\right]\right\|^2.\end{aligned} \tag{A.3}$$

By Theorem A.6, there exist constants $q_1, m_1', n_1', \overline{L}_{1,s}, s \in [0:q_1]$ that are independent of b and multi-sets of matrices $\mathcal{M}_{l,s}^1 := \left\{M_{l,s,u,v}^1 : u \in [m_1'], v \in [n_1']\right\}, s \in [q_1]$ such that

$$\mathbb{E}\left[\operatorname{tr}\left(\left(g_{t,1}^b\right)^T g_{t,1}^b\right)\middle|\mathcal{F}_0\right] = \alpha_{1,0} + \alpha_{1,1}\frac{1}{b} + \cdots + \alpha_{1,q_1}\frac{1}{b^{q_1}}, \tag{A.4}$$

where

$$\alpha_{1,s} = \sum_{l\in[\overline{L}_{1,s}]} c_{1,l,s}\operatorname{tr}\left(C_{1,l,s}\left(\bigotimes_{u\in[m_1']}\left(\prod_{v\in[n_1']} M_{l,s,u,v}^1\right)\right)\right), s \in [0:q_1],$$

$c_{1,l,s}$ is a constant, $C_{1,l,s} \in \mathcal{C}$ and $M^1_{l,s,u,v}$ only takes value in $W_0^b \bigcup W^* \bigcup \overline{\mathcal{C}}$. Further, we have $q_1 \leqslant 3^{t+1} - 1$.

It is worth mentioning that we do not include matrices $M_{1,l,s,0,v}, v \in [n_1']$ in the multi-set $\mathcal{M}^1_{l,s}, l \in [\overline{L}_{1,s}], s \in [0 : q_1]$ because each $M_{1,l,s,0,v}$ is actually an identity matrix from the proof of the previous theorems.

Similarly, there exist constants $q_2, m_2', n_2', \overline{L}_{2,s}, s \in [0 : q_2]$ that are independent of b and multi-sets of matrices $\mathcal{M}^2_{l,s} := \left\{M^2_{l,s,u,v} : u \in [0 : m_2'], v \in [n_2']\right\}, s \in [q_2]$ such that

$$\mathbb{E}\left[g^b_{t,1}\middle|\mathcal{F}_0\right] = \alpha_{2,0} + \alpha_{2,1}\frac{1}{b} + \cdots + \alpha_{2,q_2}\frac{1}{b^{q_2}}, \tag{A.5}$$

where

$$\alpha_{2,s} = \sum_{l\in[\overline{L}_{2,s}]} c_{2,l,s}\,\mathrm{tr}\left(C_{2,l,s}\left(\bigotimes_{u\in[m_2']}\left(\prod_{v\in[n_2']} M^2_{l,s,u,v}\right)\right)\right)\prod_{v\in[n_2']} M^2_{l,s,0,v}, s \in [0 : q_2],$$

$c_{2,l,s}$ is a constant, $C_{2,l,s} \in \mathcal{C}$ and $M^2_{l,s,u,v}$ only takes value in $W_0^b \bigcup W^* \bigcup \overline{\mathcal{C}}$. Further, we have $q_2 \leqslant \frac{1}{2}\left(3^{t+1} - 1\right)$.

Combining (A.3) – (A.5), we know there exist constants

$$\gamma_0, \ldots, \gamma_q, q = \max\{q_1, 2q_2\} \leqslant 3^{t+1} - 1$$

such that

$$\mathsf{var}\left(\left(W^b_{t,2}\right)^T W^b_{t,2} W^b_{t,1} x x^T\right) = \gamma_0 + \gamma_1\frac{1}{b} + \cdots \gamma_q\frac{1}{b^q},$$

∎

where

$$\gamma_s = \mathbb{E}_{W_0^t\sim\mathcal{D}'}\left[\alpha_{1,s}\right] + \sum_{u+v=s,u,v\in[0:q_2]} \mathbb{E}_{W_0^t\sim\mathcal{D}'}\left[\alpha_{2,u}\right]\mathbb{E}_{W_0^t\sim\mathcal{D}'}\left[\alpha_{2,v}\right], s \in [0 : q]$$

and $\mathcal{D}'$ is the initialization distribution of W_0^t. Further, γ_s's are independent of b.

Proof of Theorem 3.5. We first show that in $\mathsf{var}\left(g^b_{t,i}\right) = \beta_1\frac{1}{b} + \cdots + \beta_r\frac{1}{b^r}$ we have $\beta_1 \geqslant 0$. If $r = 1$, the statement obviously holds. Let us assume that the statement does not hold for $r > 1$, i.e. $\beta_1 < 0$. Taking b large enough such that $\beta_1 b^{r-1} + \beta_2 b^{r-2} + \cdots + \beta_r < 0$ yields

$$\mathsf{var}\left(g^b_{t,i}\right) = \frac{1}{b^r}\left(\beta_1 b^{r-1} + \beta_2 b^{r-2} + \cdots + \beta_r\right) < 0,$$

which contradicts the fact that $\mathsf{var}\left(g^b_{t,i}\right) \geqslant 0$. Therefore, we have $\beta_1 \geqslant 0$.

Let b_0 be large enough such that for all $b \geqslant b_0$, we have $\beta_1 b^{r-1} + 2\beta_2 b^{r-2} + \cdots + r\beta_r \geqslant 0$. We denote $f(b) = \beta_1\frac{1}{b} + \beta_2\frac{1}{b^2} + \cdots + \beta_r\frac{1}{b^r} \geqslant 0$. For all $b > b_0$ we have

$$f'(b) = -\frac{1}{b^{r+1}}\left(\beta_1 b^{r-1} + 2\beta_2 b^{r-2} + \cdots + r\beta_r\right) \leqslant 0.$$

∎

Therefore, for all $b > b_0$ we have $\left(\mathsf{var}\left(g^b_{t,i}\right)\right)' = -\frac{r}{b^{r+1}}f(b) + \frac{1}{b^r}f(b) \leqslant 0$, and thus $\mathsf{var}\left(g^b_{t,i}\right)$ is a decreasing function of b for all $b > b_0$.

A.1.2. Two-layer Networks with Quadratic Polynomial Activation Functions. In this section, we expand the scope of the theorems found in Appendix A.1.1. While they originally applied to two-layer linear networks, we now extend them to networks utilizing quadratic polynomial activation functions. The main distinction between these scenarios lies in the incorporation of Hadamard products into the gradients by the quadratic activation functions, demanding additional consideration.

Specifically, we consider a special case of the general population loss (3.1). Here the population loss is defined as $\mathcal{L}(w) = \mathbb{E}_{x\sim\mathcal{D}}\left[\frac{1}{2}\left\|W_2\sigma\left(W_1x\right) - W_2^*\sigma\left(W_1^*x\right)\right\|^2\right]$ and the SG estimators are defined as

$$g_{t,k}^b := \frac{1}{b}\sum_{i=1}^{b}\nabla_{W_{t,k}^b}\left(\frac{1}{2}\left\|W_{t,2}^b\sigma\left(W_{t,1}^bx_{t,i}^b\right) - W_2^*\sigma\left(W_1^*x_{t,i}^b\right)\right\|^2\right), \quad k = 1, 2,$$

where $\sigma(x) := \sigma_0 + \sigma_1 x + \sigma_2 x^2$ is a polynomial activation function of degree 2. This setup aligns to the $D = 2$ and $H = 2$ case as in (3.1).

Similar to (A.1) – (A.2), we rewrite the SG estimator as the sum of the product of weight matrices and other constant matrices. For example, we have

$$\begin{aligned} g_{t,1}^b &= \frac{1}{b}\sum_{i=1}^{b}\nabla_{W_{t,1}^b}\left(\frac{1}{2}\left\|W_{t,2}^b\sigma\left(W_{t,1}^bx_{t,i}^b\right) - W_2^*\sigma\left(W_1^*x_{t,i}^b\right)\right\|^2\right) \\ &= \frac{1}{2b}\sum_{i=1}^{b}\nabla_{W_{t,1}^b}\Big\|\sigma_2W_{t,2}^b\left(\left(W_{t,1}^bx_{t,i}^b\right)\odot\left(W_{t,1}^bx_{t,i}^b\right)\right) + \sigma_1W_{t,2}^b\left(W_{t,1}^bx_{t,i}^b\right) + \sigma_0W_{t,2}^b \\ &\qquad - \sigma_2W_2^*\left(\left(W_1^bx_{t,i}^b\right)\odot\left(W_1^*x_{t,i}^b\right)\right) - \sigma_1W_2^*\left(W_1^*x_{t,i}^b\right) - \sigma_0W_2^*\Big\|^2. \end{aligned} \tag{A.6}$$

We first show how to calculate the gradient of a mixed form with common and Hadamard products. With this approach, we can represent each summand of (A.6) as a summation of terms in the form of $\prod_k M_k$, where M_k or its transpose only takes on values from $\{W_{t,1}^b, W_{t,2}^b, W_1^*, W_2^*, x_{t,i}^b\}\bigcup\mathcal{C}$.

We take two terms in the expansion of the summand in (A.6) as examples to show how to replace the Hadamard products by common products. We use the fact that, for any positive integer n and vectors $v_1, \ldots, v_n \in \mathbb{R}^p$,

$$v_1\odot v_2\odot\cdots\odot v_n = \sum_{j\in p}\left(e_{p,j}^Tv_1\right)\left(e_{p,j}^Tv_2\right)\cdots\left(e_{p,j}^Tv_n\right)e_{p,j}, \tag{A.7}$$

where $e_{p,j}, j\in[p]$ is the j-th unit vector in $\mathbb{R}^p$.

For example, we have[4]

$$\begin{aligned} &\nabla_{W_{t,1}^b}\operatorname{tr}\left(\sigma_1\left(W_{t,1}^bx_{t,i}^b\right)^T\left(W_{t,2}^b\right)^T\sigma_2W_{t,2}^b\left(\left(W_{t,1}^bx_{t,i}^b\right)\odot\left(W_{t,1}^bx_{t,i}^b\right)\right)\right) \\ &= \sigma_1\sigma_2\sum_{j\in[p_1]}\nabla_{W_{t,1}^b}\operatorname{tr}\left(\left(x_{t,i}^b\right)^T\left(W_{t,1}^b\right)^T\left(W_{t,2}^b\right)^TW_{t,2}^b\left(e_{p_1,j}^TW_{t,1}^bx_{t,i}^b\right)\left(e_{p_1,j}^TW_{t,1}^bx_{t,i}^b\right)e_{p_1,j}\right) \\ &= \sigma_1\sigma_2\sum_{j\in[p_1]}\Big[\left(W_{t,2}^b\right)^TW_{t,2}^be_{p_1,j}^TW_{t,1}^bx_{t,i}^be_{p_1,j}^TW_{t,1}^bx_{t,i}^be_{p_1,j}\left(x_{t,i}^b\right)^T \\ &\qquad + e_{p_1,j}\left(W_{t,2}^b\right)^TW_{t,2}^bW_{t,1}^bx_{t,i}^be_{p_1,j}^T\left(x_{t,i}^b\right)^T\left(W_{t,1}^b\right)^Te_{p_1,j}\left(x_{t,i}^b\right)^T \\ &\qquad + e_{p_1,j}\left(x_{t,i}^b\right)^T\left(W_{t,1}^b\right)^Te_{p_1,j}\left(W_{t,2}^b\right)^TW_{t,2}^bW_{t,1}^bx_{t,i}^be_{p_1,j}^T\left(x_{t,i}^b\right)^T\Big] \end{aligned}$$

[4]We frequently use the fact, that for matrices A, B, X with appropriate dimensions, $\nabla_X \operatorname{tr}(AXB) = A^TB^T$ and $\nabla_X \operatorname{tr}(AX^TB) = BA$.

and

$$\begin{aligned}
&\nabla_{W_{t,1}^b} \operatorname{tr}\left(\sigma_2 \left[W_{t,2}^b \left(\left(W_{t,1}^b x_{t,i}^b\right) \odot \left(W_{t,1}^b x_{t,i}^b\right)\right)\right]^T \sigma_2 W_{t,2}^b \left(\left(W_{t,1}^b x_{t,i}^b\right) \odot \left(W_{t,1}^b x_{t,i}^b\right)\right)\right) \\
\text{(A.8)}& \\
&= \sigma_2^2 \sum_{j,k\in[p_1]} \nabla_{W_{t,1}^b} \operatorname{tr}(e_{p_1,k}^T \left(x_{t,i}^b\right)^T \left(W_{t,1}^b\right)^T e_{p_1,k} \left(x_{t,i}^b\right)^T \left(W_{t,1}^b\right)^T \cdot \\
&\qquad \cdot e_{p_1,k} \left(W_{t,2}^b\right)^T W_{t,2}^b e_{p_1,j}^T W_{t,1}^b x_{t,i}^b e_{p_1,j}^T W_{t,1}^b x_{t,i}^b e_{p_1,j}) \\
&= \sigma_2^2 \sum_{j,k\in[p_1]} \Bigg[e_{p_1,k} \left(x_{t,i}^b\right)^T \left(W_{t,1}^b\right)^T e_{p_1,k} \left(W_{t,2}^b\right)^T W_{t,2}^b e_{p_1,j}^T W_{t,1}^b x_{t,i}^b e_{p_1,j}^T W_{t,1}^b x_{t,i}^b e_{p_1,j} e_{p_1,k}^T \left(x_{t,i}^b\right)^T \\
&\qquad + e_{p_1,k} \left(W_{t,2}^b\right)^T W_{t,2}^b e_{p_1,j}^T W_{t,1}^b x_{t,i}^b e_{p_1,j}^T W_{t,1}^b x_{t,i}^b e_{p_1,j} e_{p_1,k}^T \left(x_{t,i}^b\right)^T \left(W_{t,1}^b\right)^T e_{p_1,k} \left(x_{t,i}^b\right)^T \\
&\qquad + e_{p_1,j} \left(W_{t,2}^b\right)^T W_{t,2}^b e_{p_1,k}^T W_{t,1}^b x_{t,i}^b e_{p_1,k}^T W_{t,1}^b x_{t,i}^b e_{p_1,k} e_{p_1,j}^T \left(x_{t,i}^b\right)^T \left(W_{t,1}^b\right)^T e_{p_1,j} \left(x_{t,i}^b\right)^T \\
\text{(A.9)}& \\
&\qquad + e_{p_1,j} \left(x_{t,i}^b\right)^T \left(W_{t,1}^b\right)^T e_{p_1,j} \left(W_{t,2}^b\right)^T W_{t,2}^b e_{p_1,k}^T W_{t,1}^b x_{t,i}^b e_{p_1,k}^T W_{t,1}^b x_{t,i}^b e_{p_1,k} e_{p_1,j}^T \left(x_{t,i}^b\right)^T \Bigg].
\end{aligned}$$

In conclusion, there exist constants $J, K, \alpha_j, j \in [J]$ independent of b and a multi-set of matrices $\{M_{s,i,j,k}, i \in [b], j \in [J], k \in [K], s = 1, 2\}$ such that

$$g_{t,s}^b = \frac{1}{b} \sum_{i\in[b]} \sum_{j\in[J]} \left(\alpha_{s,i,j} \prod_{k\in[K]} M_{s,i,j,k}\right), s = 1, 2,$$

where $M_{s,i,j,k}$ or its transpose only takes value in $\{W_{t,1}^b, W_{t,2}^b, W_1^*, W_2^*\} \bigcup \{x_{t,i}^b, i \in [b]\} \bigcup \mathcal{C}$.

It is worth mentioning that we can provide the exact values of J and K, namely $J = 144p_1^2$ and $K = 15$. These numbers are determined by analyzing the most complicated term, i.e. the left-hand side of (A.9), among the expansion of summands in (A.6). Note that the summation on the right-hind side of (A.9) contributes $4p_1^2$ terms where each term is a product of 15 matrices and the expansion of a summand in (A.6) gives 36 terms of matrices' mixed products. Thus we have $J = 36 \cdot 4p_1^2 = 144p_1^2$ and $K = 15$. We can use identity matrices and zeros to fill up the unused $M_{s,i,j,k}$ and $\alpha_{s,i,j}$ as needed.

This representation aligns with the right-hand side of (A.1) and (A.2), excepts the fact that we further expand the $\mathcal{W}_t^b = W_{t,2}^b W_{t,1}^b - W_2^* W_1^*$ to separate terms. Thus we can further analyze the dynamics of polynomially-activated networks in a similar manner as in Appendix A.1.1.

A.1.3. Deep Networks with Polynomially-activated Functions. In this section, we discuss the extension from two-layer network networks with quadratic polynomial activation functions to deep networks with polynomial activation functions of any degree. In other words, we consider the general setting where D and H can take arbitrary values as in (3.1).

The building block of above derivation is to represent the SG estimators as products of weights matrices, samples, and other constant matrices. However, given the arbitrary values of D and H, the number of matrices required is much more than the case as in Appendix A.1.2.

LEMMA A.8. *There exist constants $J, K, \alpha_j, j \in [J]$ independent of b and a multi-*

set of matrices $\{M_{s,i,j,k}, i \in [b], j \in [J], k \in [K], s \in [H]\}$ *such that, for any* $s \in [H]$,

$$g^b_{t,s} := \frac{1}{b}\sum_{i=1}^{b} \nabla_{W^b_{t,s}}\left(\frac{1}{2}\left\|W^b_{t,H}\sigma\left(W^b_{t,H-1}\sigma\left(\cdots\sigma\left(W^b_{t,1}x^b_{t,i}\right)\right)\right) - W^*_H\sigma\left(W^*_{H-1}\sigma\left(\cdots\sigma\left(W^*_1 x^b_{t,i}\right)\right)\right)\right\|^2\right)$$
$$= \frac{1}{b}\sum_{i\in[b]}\sum_{j\in[J]}\left(\alpha_{s,i,j}\prod_{k\in[K]} M_{s,i,j,k}\right), \tag{A.10}$$

where $M_{s,i,j,k}$ *or its transpose only takes value in* $\{W^b_{t,1}, W^b_{t,2}, W^*_1, W^*_2\}\bigcup\{x^b_{t,i}, i \in [b]\}\bigcup\mathcal{C}$.

To give an insight on the complexity of this representation, we provide the possible values of J and K[5] in an induction fashion.

- $K = 6D^{H-1} + 4D^{H-2} + \cdots + 4D + 3$.
  In the expansion of $W^b_{t,2}\sigma\left(W^b_{t,1}x^b_{t,i}\right)$, the most complicated term[6] is $W^b_{t,2}\big(W^b_{t,1} x^b_{t,i}\big)^{\odot D}$. By applying (A.7), we can rewrite it as a sum of product of $3D+2$ matrices, namely $\sum_{j_1\in[p_1]} W^b_{t,2}\left(e^T_{p_1,j_1}W^b_{t,1}x^b_{t,i}\right)^D e_{p_1,j_1}$. Similarly, the most complicated term in the expansion of $W^b_{t,3}\sigma\left(W^b_{t,2}\sigma\left(W^b_{t,1}x^b_{t,i}\right)\right)$ is a sum of product of $D(3D+2)+2 = 3D^2+2D+2$ matrices, namely
  $$\sum_{j_2}\left(e^T_{p_2,j_2}\left(\sum_{j_1} W^b_{t,2}\left(e^T_{p_1,j_1}W^b_{t,1}x^b_{t,i}e_{p_1,j_1}\right)^D\right)\right)^D e_{p_2,j_2}.$$
  We can use induction to prove that the number of matrices needed for layer s should be D times the number of matrices needed for layer $s-1$ plus 2. For a general H-layer network, we require $\overline{K} := 3D^{H-1}+2D^{H-2}+\cdots+2D+2$ matrices to represent the most complicated term in $W^b_{t,H}\sigma\left(W^b_{t,H-1}\sigma\left(\cdots\sigma\left(W^b_{t,1}x^b_{t,i}\right)\right)\right)$. Thus we set $K = 2\overline{K}-1 = 6D^{H-1}+4D^{H-2}+\cdots+4D+3$ due to the square operator in the norm and minus one by taking the gradient with respect to $W^b_{t,s}$.
- $J = \left[2\left(D^{H-1}+\cdots+D+1\right)p_1^{H-1}p_2^{H-2}\cdots p_{H-1}D^{H-1}\right]^2$
  From the derivation above, we can see that the, in the expansion of
  $$W^b_{t,H}\sigma\left(W^b_{t,H-1}\sigma\left(\cdots\sigma\left(W^b_{t,1}x^b_{t,i}\right)\right)\right),$$
  the most complicated term consists of $p_1^{H-1}p_2^{H-2}\cdots p_{H-1}$ terms of product of matrices and $W^b_{t,1}$ appears most frequently in each of these products (D^{H-1} times). Besides, as there are in total of $D^{H-1}+\cdots+D+1$ terms if simply replace the activation function σ by the equivalent polynomial, we end up with $2\left(D^{H-1}+\cdots+D+1\right)p_1^{H-1}p_2^{H-2}\cdots p_{H-1}D^{H-1}$ terms for $W^b_{t,H}\sigma\left(W^b_{t,H-1}\sigma\left(\cdots\sigma\left(W^b_{t,1}x^b_{t,i}\right)\right)\right) - W^*_H\sigma\left(W^*_{H-1}\sigma\left(\cdots\sigma\left(W^*_1x^b_{t,i}\right)\right)\right)$. By taking the square, we expect
  $$J = \left[2\left(D^{H-1}+\cdots+D+1\right)p_1^{H-1}p_2^{H-2}\cdots p_{H-1}D^{H-1}\right]^2.$$

Again, the representation in (A.10) aligns with the right-hand side of (A.1) and (A.2). Thus we can further analyze the dynamics of polynomially-activated networks in a similar manner as in Appendix A.1.1.

[5]As we can always padding identity matrices to $M_{s,i,j,k}$, thus the values of J and K are not unique.

[6]We ignore the constant coefficient σ_D here for convenience.

A.1.4. Deep Networks with General Activation Functions. In this section, we discuss the extension from a polynomially-activated network to a neural network with general activation functions under mild assumptions. Given a neural network $f^S(x) := W_H^S \sigma^S \left(W_{H-1}^S \cdots \sigma^S \left(W_1^S x\right)\right)$ with the population loss $\mathcal{L}(w^S) = \mathbb{E}_{x\sim\mathcal{D}}\left[\frac{1}{2}\left\|W_H^S \sigma^S \left(W_{H-1}^S \cdots \sigma^S \left(W_1^S x\right)\right) - W_H^* \sigma^S \left(W_{H-1}^* \cdots \sigma^S \left(W_1^* x\right)\right)\right\|^2\right]$, we define the gradient corresponding to each sample $x_{t,i}, i \in [b]$ and $k \in [H]$ as[7]

$$g_{t,k,i}^S := \nabla_{W_{t,k}^S}\left(\frac{1}{2}\left\|W_{t,H}^S \sigma^S \left(W_{t,H-1}^S \cdots \sigma^S \left(W_{t,1}^S x_{t,i}\right)\right) - W_H^* \sigma^S \left(W_{H-1}^* \cdots \sigma^S \left(W_1^* x_{t,i}\right)\right)\right\|^2\right).$$

Following Section 3.1 of [40], we define a set of intermediate variables

$$\begin{aligned}
z_{t,0,i}^S &= x_{t,i}, & h_{t,1,i}^S &= W_{t,1}^S z_{t,0,i}^S,\\
z_{t,1,i}^S &= \sigma^S\left(h_{t,1,i}^S\right), & h_{t,2,i}^S &= W_{t,2}^S z_{t,1,i}^S,\\
&\vdots, & &\vdots\\
z_{t,H-1,i}^S &= \sigma^S\left(h_{t,H-1,i}^S\right), & h_{t,H,i}^S &= W_{t,H}^S z_{t,H-1,i}^S,
\end{aligned}$$

and $D_{t,k,i}^S = \operatorname{diag}\left(\sigma_S'\left(h_{t,k,i}^S\right)\right)$, where σ_S' represents the derivative of the activation function σ^S and $\operatorname{diag}(v)$ maps a vector v to its corresponding diagonal representation. The SG estimators over weight matrix $W_{t,k}^S$ are given by

$$\begin{aligned}
g_{t,k}^S := \frac{1}{b}\sum_{i\in[b]} g_{t,k,i}^S = \frac{1}{b}\sum_{i\in[b]} W_{t,H}^S D_{t,H-1,i}^S \cdots W_{t,k+2}^S D_{t,k+1,i}^S W_{t,k+1}^S D_{t,k,i}^S \cdot\\
\cdot\left[W_{t,H}^S \sigma^S \left(W_{t,H-1}^S \cdots \sigma^S \left(W_{t,1}^S x_{t,i}\right)\right) - W_H^* \sigma^S \left(W_{H-1}^* \cdots \sigma^S \left(W_1^* x_{t,i}\right)\right)\right]\left(z_{t,k-1,i}^S\right)^T.
\end{aligned}$$

We further assume that

- σ^S is smooth on $\mathbb{R}^p$,
- $\|x_{t,i}\|$ is bounded, i.e., there exists a constant C_x such that $\|x_{t,i}\| \leqslant C_x, \forall t \in [T], i \in [b]$,
- $\left\|W_{t,k}^S\right\|$ is bounded, i.e., there exists a constant C_W such that $\left\|W_{t,k}^S\right\| \leqslant C_W$,
- $\left\|h_{t,k,i}^S\right\|$ is bounded, i.e., there exists a constant C_h such that $\left\|h_{t,k,i}^S\right\| \leqslant C_h$. [8]

We denote $\mathcal{R} := [-C_h, C_h]^p$. By the first assumption, there exists a constant C_S such that $\left\|\sigma^S(x)\right\| \leqslant C_s, \forall x \in \mathcal{R}$. Note that $\|h_{t,k,i}\|_\infty \leqslant \|h_{t,k,i}\| \leqslant C_h$, thus $h_{t,k,i} \in \mathcal{R}$ for all $t \in [T], k \in [H], i \in [b]$.

We note that these assumptions hold in several of the neural network training regimes. For example, the Sigmoid function meets the first assumption with $C_S = 1$, $\mathcal{R} = [-C_h, C_h]^p$ for $C_h = C_h(C_W, C_x) < \infty$, and both Sigmoid function and its derivative are Lipschitz continuous.

Similarly, we define a polynomially-activated neural network $f^P(x) := W_H^P \sigma^P \big(W_{H-1}^P \cdots \sigma^P \left(W_1^P x\right)\big)$ where $\sigma^P(\cdot)$ is a polynomial function. The loss function and SG estimators are defined similarly except for switching the superscript S to P. We

[7] For simplicity, we remove the superscript b in this section.

[8] In fact, C_h can be expressed as a function of C_W, C_x, and $\left\|\sigma^S(\cdot)\right\|$. For example, taking $C_{S,0} = C_x$ and we further find a constant $C_{S,k}$ such that $\left\|\sigma^S(x)\right\| \leqslant C_{S,k}$ holds for all $\|x\| \leqslant C_W C_{S,k-1}, k \in [H-1]$, then we have $h_{t,k,i}^S = W_{t,k}^S \sigma^S\left(h_{t,k-1,i}^S\right) \leqslant C_W C_{S,k}$. Taking $C_h = C_W \max_{k\in[H]}\left\{C_{S,k}\right\}$ satisfies the assumption.

use SGD to optimize the loss of these two neural networks with the same initial points ($W_{0,k} := W^S_{0,k} = W^P_{0,k}, k \in [H]$), ground-truth weights ($W^*_1, \ldots, W^*_H$), samples ($x_{t,i}, i \in [b]$), and learning rate α_t in every iteration.

In the following, we show that, if the polynomial σ^P is a good approximation of the activation function σ^S over $\overline{\mathcal{R}}$[9], then the SG estimators $g^S_{t,k}$ and $g^P_{t,k}, k \in [H]$ are also close enough. Formally, we have

THEOREM A.9. *For any $\epsilon > 0$ and time step $T \in \mathbb{N}^+$, there exists a polynomial $\sigma^P(\cdot)$ (depending on ϵ, σ^S, and T) such that $\left\| g^S_{T,k} - g^P_{T,k} \right\| \leqslant \epsilon, k \in [H]$.*

Outline of the Proof. We choose a polynomial function σ^P such that $\|\sigma^S(x) -\sigma^P(x)\| \leqslant \epsilon'$ and $\|\sigma'_S(x) - \sigma'_P(x)\| \leqslant \epsilon'$ both hold over $\overline{\mathcal{R}} := [-2C_h, 2C_h]^p$ and $\mathcal{O}(\epsilon') < C_h$. The exact value of $\epsilon' < 1$ is determined later[10]. In the following, we induct on t to show that

(1) $\left\| W^S_{t,k} - W^P_{t,k} \right\| \leqslant \mathcal{O}(\epsilon'), k \in [H]$,

(2) $\left\| h^S_{t,k,i} - h^P_{t,k,i} \right\| \leqslant \mathcal{O}(\epsilon'), k \in [H], i \in [b]$,

(3) $\left\| z^S_{t,k,i} - z^P_{t,k,i} \right\| \leqslant \mathcal{O}(\epsilon'), k \in [H], i \in [b]$,

(4) $\left\| D^S_{t,k,i} - D^P_{t,k,i} \right\| \leqslant \mathcal{O}(\epsilon'), k \in [H], i \in [b]$,

(5) $h^P_{t,k,i} \in \overline{\mathcal{R}}, k \in [H], i \in [b]$,

(6) $\left\| g^S_{t,k} - g^P_{t,k} \right\| \leqslant \mathcal{O}(\epsilon'), k \in [H]$,

where $\mathcal{O}(\cdot)$ is used to hide constants that relate to $L_S, L'_S, C_S, C_W, C_h, C_x, d_k, k \in [H]$ and are independent of ϵ'. In the following, we use $(1)_t, \ldots, (5)_t$ to represent the statements at time step t, respectively. For (2), (3), (4), and (5), we use $(2)_{t,k}, \ldots, (5)_{t,k}$ to specify the statements for the k-th layer at time step t, respectively.

For $t = 0$, $(1)_t$ is obvious since $W^S_{0,k} = W^P_{0,k}, k \in [H]$.

For $t \geqslant 0, (1)_t \Rightarrow (2)_t, (3)_t, (4)_t$, we further induct on k to prove them for any given t.

- $k = 1, (1)_t \Rightarrow (2)_{t,1}$

$$\left\| h^S_{t,1,i} - h^P_{t,1,i} \right\| = \left\| W^S_{t,1} z^S_{t,0,i} - W^P_{t,1} z^P_{t,0,i} \right\| \leqslant \left\| W^S_{t,1} - W^P_{t,1} \right\| \|x_{t,i}\| \leqslant \mathcal{O}\left(\epsilon'\right) C_x = \mathcal{O}\left(\epsilon'\right).$$

- $k \in [H], (2)_{t,k} \Rightarrow (5)_{t,k}$

$$\left\| h^P_{t,k,i} \right\|_\infty \leqslant \left\| h^P_{t,k,i} \right\| \leqslant \left\| h^S_{t,k,i} - h^P_{t,k,i} \right\| + \left\| h^S_{t,k,i} \right\| \leqslant \mathcal{O}\left(\epsilon'\right) + C_h \leqslant 2C_h$$

- $k \in [H-1], (2)_{t,k}, (5)_{t,k} \Rightarrow (3)_{t,k}$

$$\begin{aligned}\left\| z^S_{t,k,i} - z^P_{t,k,i} \right\| &= \left\| \sigma^S\left(h^S_{t,k,i}\right) - \sigma^P\left(h^P_{t,k,i}\right) \right\| \leqslant \left\| \sigma^S\left(h^S_{t,k,i}\right) - \sigma^P\left(h^S_{t,k,i}\right) \right\| + \left\| \sigma^P\left(h^S_{t,k,i}\right) - \right. \\ &\left. - \sigma^P\left(h^P_{t,k,i}\right) \right\| \leqslant \epsilon' + L_P \left\| h^S_{t,k,i} - h^P_{t,k,i} \right\| \leqslant \epsilon' + L_P \mathcal{O}\left(\epsilon'\right) = \mathcal{O}\left(\epsilon'\right)\end{aligned}$$

[9]The rigorous definition of $\overline{\mathcal{R}}$ is provided in the proof.

[10]Note that this polynomial is guaranteed to exist since the general activation function σ^S is continuous over the compact domain $\overline{\mathcal{R}}$.

- $k \in [2:H], (3)_{t,k-1} \Rightarrow (2)_{t,k}$

$$\begin{aligned}\left\|h_{t,k,i}^S - h_{t,k,i}^P\right\| &= \left\|W_{t,k}^S z_{t,k-1,i}^S - W_{t,k}^P z_{t,k-1,i}^P\right\| \\ &= \left\|W_{t,k}^S z_{t,k-1,i}^S - W_{t,k}^P z_{t,k-1,i}^S + W_{t,k}^P z_{t,k-1,i}^S - W_{t,k}^P z_{t,k-1,i}^P\right\| \\ &\leqslant \left\|W_{t,k}^S - W_{t,k}^P\right\| \left\|z_{t,k-1,i}^S\right\| + \left\|W_{t,k}^P\right\| \left\|z_{t,k-1,i}^S - z_{t,k-1,i}^P\right\| \\ &\leqslant \mathcal{O}\left(\epsilon'\right) \left\|\sigma^S\left(h_{t,k-1,i}^S\right)\right\| + \left(\left\|W_{t,k}^P - W_{t,k}^S\right\| + \left\|W_{t,k}^S\right\|\right) \mathcal{O}\left(\epsilon'\right) \\ &\leqslant C_S \mathcal{O}\left(\epsilon'\right) + \left(\mathcal{O}\left(\epsilon'\right) + C_W\right) \mathcal{O}\left(\epsilon'\right) \leqslant \mathcal{O}\left(\epsilon'\right)\end{aligned}$$

- $k \in [H], (2)_{t,k} \Rightarrow (4)_{t,k}$

$$\begin{aligned}\left\|D_{t,k,i}^S - D_{t,k,i}^P\right\| &= \left\|\operatorname{diag}\left(\sigma_S'\left(h_{t,k,i}^S\right)\right) - \operatorname{diag}\left(\sigma_P'\left(h_{t,k,i}^P\right)\right)\right\| \\ &= \left\|\sigma_S'\left(h_{t,k,i}^S\right) - \sigma_P'\left(h_{t,k,i}^P\right)\right\|_\infty \leqslant \left\|\sigma_S'\left(h_{t,k,i}^S\right) - \sigma_P'\left(h_{t,k,i}^P\right)\right\| \\ &\leqslant \left\|\sigma_S'\left(h_{t,k,i}^S\right) - \sigma_P'\left(h_{t,k,i}^S\right)\right\| + \left\|\sigma_P'\left(h_{t,k,i}^S\right) - \sigma_P'\left(h_{t,k,i}^P\right)\right\| \\ &\leqslant \epsilon' + L_P' \left\|h_{t,k,i}^S - h_{t,k,i}^P\right\| \leqslant \epsilon' + L_P' \mathcal{O}\left(\epsilon'\right) = \mathcal{O}\left(\epsilon'\right)\end{aligned}$$

For $t \geqslant 0, (1)_t + \cdots + (5)_t \Rightarrow (6)_t$, we denote $h_{t,i}^{S*} := W_H^* \sigma^S\left(W_{H-1}^* \cdots \sigma^S\left(W_t^* x_{t,i}\right)\right)$ and $h_{t,i}^{P*} := W_H^* \sigma^P\left(W_{H-1}^* \cdots \sigma^P\left(W_t^* x_{t,i}\right)\right)$. Note that

$$\left\|g_{t,k}^S - g_{t,k}^P\right\| = \left\|\frac{1}{b}\sum_{i\in[b]} g_{t,k,i}^S - \frac{1}{b}\sum_{i\in[b]} g_{t,k,i}^P\right\| \leqslant \frac{1}{b}\sum_{i\in[b]}\left\|g_{t,k,i}^S - g_{t,k,i}^P\right\|.$$

For each $i \in [b]$, we have

$$\begin{aligned}&\left\|g_{t,k,i}^S - g_{t,k,i}^P\right\| \\ &= \Big\|W_{t,H}^S D_{t,H-1,i}^S \cdots W_{t,k+1}^S D_{t,k,i}^S \left(h_{t,H,i}^S - h_{t,i}^{S*}\right)\left(z_{t,k-1,i}^S\right)^T \\ &\quad - W_{t,H}^P D_{t,H-1,i}^P \cdots W_{t,k+1}^P D_{t,k,i}^P \left(h_{t,H,i}^P - h_{t,i}^{P*}\right)\left(z_{t,k-1,i}^P\right)^T\Big\| \\ &\leqslant \Big\|W_{t,H}^S D_{t,H-1,i}^S \cdots W_{t,k+1}^S D_{t,k,i}^S h_{t,H,i}^S \left(z_{t,k-1,i}^S\right)^T \\ &\quad - W_{t,H}^P D_{t,H-1,i}^P \cdots W_{t,k+1}^P D_{t,k,i}^P h_{t,H,i}^P \left(z_{t,k-1,i}^P\right)^T\Big\| + \\ &\quad + \Big\|W_{t,H}^S D_{t,H-1,i}^S \cdots W_{t,k+1}^S D_{t,k,i}^S h_{t,i}^{S*} \left(z_{t,k-1,i}^S\right)^T \\ &\quad\quad - W_{t,H}^P D_{t,H-1,i}^P \cdots W_{t,k+1}^P D_{t,k,i}^P h_{t,i}^{P*} \left(z_{t,k-1,i}^P\right)^T\Big\|. \qquad \text{(A.11)}\end{aligned}$$

For the first item in (A.11), we have

$$
\begin{aligned}
&\left\| W^S_{t,H} D^S_{t,H-1,i} \cdots W^S_{t,k+1} D^S_{t,k,i} h^S_{t,H,i} \left(z^S_{t,k-1,i} \right)^T - W^P_{t,H} D^P_{t,H-1,i} \cdots W^P_{t,k+1} D^P_{t,k,i} h^P_{t,H,i} \left(z^P_{t,k-1,i} \right)^T \right\| \\
=& \left\| W^S_{t,H} D^S_{t,H-1,i} \cdots W^S_{t,k+1} D^S_{t,k,i} W^S_{t,H} z^S_{t,H-1,i} \left(z^S_{t,k-1,i} \right)^T - \right. \\
&\quad \left. - W^P_{t,H} D^P_{t,H-1,i} \cdots W^P_{t,k+1} D^P_{t,k,i} W^P_{t,H} z^P_{t,H-1,i} \left(z^P_{t,k-1,i} \right)^T \right\| \\
\leqslant& \left\| W^S_{t,H} D^S_{t,H-1,i} \cdots W^S_{t,k+1} D^S_{t,k,i} W^S_{t,H} z^S_{t,H-1,i} - \right. \\
&\quad \left. - W^P_{t,H} D^P_{t,H-1,i} \cdots W^P_{t,k+1} D^P_{t,k,i} W^P_{t,H} z^P_{t,H-1,i} \right\| \left\| z^P_{t,k-1,i} \right\| + \\
&+ \left\| W^S_{t,H} D^S_{t,H-1,i} \cdots W^S_{t,k+1} D^S_{t,k,i} W^S_{t,H} z^S_{t,H-1,i} \right\| \left\| z^S_{t,k-1,i} - z^P_{t,k-1,i} \right\| \\
\leqslant& \left\| W^S_{t,H} D^S_{t,H-1,i} \cdots W^S_{t,k+1} D^S_{t,k,i} W^S_{t,H} z^S_{t,H-1,i} - \right. \\
&\quad \left. - W^P_{t,H} D^P_{t,H-1,i} \cdots W^P_{t,k+1} D^P_{t,k,i} W^P_{t,H} z^P_{t,H-1,i} \right\| \cdot \sqrt{d_{k-1}} \left\| z^P_{t,k-1,i} \right\|_\infty + \\
&+ \left\| W^S_{t,H} \right\| \left\| D^S_{t,H-1,i} \right\| \cdots \left\| W^S_{t,k+1} \right\| \left\| D^S_{t,k,i} \right\| \left\| W^S_{t,H} \right\| \left\| z^S_{t,H-1,i} \right\| \mathcal{O}\left(\epsilon'\right) \\
\leqslant& \left\| W^S_{t,H} D^S_{t,H-1,i} \cdots W^S_{t,k+1} D^S_{t,k,i} W^S_{t,H} z^S_{t,H-1,i} - \right. \\
&\quad \left. - W^P_{t,H} D^P_{t,H-1,i} \cdots W^P_{t,k+1} D^P_{t,k,i} W^P_{t,H} z^P_{t,H-1,i} \right\| \cdot \sqrt{d_{k-1}} C_S + \\
&+ C_W^{H-k+1} C_S'^{H-k} \sqrt{d_{H-1}} C_S \mathcal{O}\left(\epsilon'\right) \\
\leqslant& \left\| W^S_{t,H} D^S_{t,H-1,i} \cdots W^S_{t,k+1} D^S_{t,k,i} W^S_{t,H} - W^P_{t,H} D^P_{t,H-1,i} \cdots W^P_{t,k+1} D^P_{t,k,i} W^P_{t,H} \right\| \left\| z^P_{t,H-1,i} \right\| \cdot \sqrt{d_{k-1}} C_S \\
&+ \left\| W^S_{t,H} D^S_{t,H-1,i} \cdots W^S_{t,k+1} D^S_{t,k,i} W^S_{t,H} z^S_{t,H-1,i} \right\| \left\| z^S_{t,H-1,i} - z^P_{t,H-1,i} \right\| \sqrt{d_{k-1}} C_S + \mathcal{O}\left(\epsilon'\right) \\
=& \left\| W^S_{t,H} D^S_{t,H-1,i} \cdots W^S_{t,k+1} D^S_{t,k,i} W^S_{t,H} - \right. \\
&\quad \left. - W^P_{t,H} D^P_{t,H-1,i} \cdots W^P_{t,k+1} D^P_{t,k,i} W^P_{t,H} \right\| \left\| z^P_{t,H-1,i} \right\| \cdot \sqrt{d_{k-1}} C_S + \mathcal{O}\left(\epsilon'\right) + \mathcal{O}\left(\epsilon'\right) \\
\leqslant& \left\| W^S_{t,H} D^S_{t,H-1,i} \cdots W^S_{t,k+1} D^S_{t,k,i} W^S_{t,H} - \right. \\
&\quad \left. - W^P_{t,H} D^P_{t,H-1,i} \cdots W^P_{t,k+1} D^P_{t,k,i} W^P_{t,H} \right\| \cdot \sqrt{d_{H-1} d_{k-1}} C_S^2 + \mathcal{O}\left(\epsilon'\right) \\
\leqslant& \cdots\cdots \\
\leqslant& \mathcal{O}\left(\epsilon'\right).
\end{aligned}
$$

Similarly, we can show that the second term in (A.11) is also bounded by $\mathcal{O}\left(\epsilon'\right)$. Thus we have $\left\| g^S_{t,k} - g^P_{t,k} \right\| \leqslant \frac{1}{b} \sum_{i \in [b]} \left\| g^S_{t,k,i} - g^P_{t,k,i} \right\| \leqslant \mathcal{O}\left(\epsilon'\right)$.

For $t \geqslant 0, (1)_t + (5)_t \Rightarrow (1)_{t+1}$, we have

$$
\begin{aligned}
\left\| W^S_{t+1,k} - W^P_{t+1,k} \right\| &= \left\| \left(W^S_{t,k} - \alpha_t g^S_{t,k} \right) - \left(W^P_{t,k} - \alpha_t g^P_{t,k} \right) \right\| \\
&\leqslant \left\| W^S_{t,k} - W^P_{t,k} \right\| + \alpha_t \left\| g^S_{t,k} - g^P_{t,k} \right\| \leqslant \mathcal{O}\left(\epsilon'\right) + \alpha_t \mathcal{O}\left(\epsilon'\right) = \mathcal{O}\left(\epsilon'\right).
\end{aligned}
$$

With the above steps, we have finished the induction. The proof is achieved by taking ϵ' small enough such that $\mathcal{O}(\epsilon') < \epsilon$ at time step T. ∎

While the above theorem only discuss the closeness of $g^S_{T,k}$ and $g^P_{T,k}$, it is worth mentioning that the same statement holds for all pairs of intermediate variables or even composition of them. In fact, we have the following generalized theorem.

THEOREM A.10. *For any $\epsilon > 0$ and time step $T \in \mathbb{N}^+$, there exists a polynomial $\sigma^P(\cdot)$ (depending on ϵ, σ^S, and T) such that*

$$
\left\| \operatorname{tr}\left(C \left(\bigotimes_i \prod_j M^S_{i,j} \right) \right) \prod_j M^S_{0,j} - \operatorname{tr}\left(C \left(\bigotimes_i \prod_j M^P_{i,j} \right) \right) \prod_j M^P_{0,j} \right\| < \epsilon,
$$

where $M^S_{i,j}$ *takes values in* $W^S_{0:t} \bigcup G^S_{0:T} \bigcup W^* \bigcup \overline{\mathcal{C}}$ *and* $M^P_{i,j}$ *takes the corresponding variable in the polynomially-activated network as of* $M^S_{i,j}$.

Together with the closed-form representation of $\mathrm{tr}\left(C\left(\bigotimes_i \prod_j M^P_{i,j}\right)\right)\prod_j M^P_{0,j}$, we are able to provide an approximation of $\mathrm{tr}\left(C\left(\bigotimes_i \prod_j M^S_{i,j}\right)\right)\prod_j M^S_{0,j}$ at any time step T with any precision. In other words, we have provided an approximation for a generalized form of mixed product at time step t using solely the initial weights W^b_0 and other constant matrices. Similarly, Theorem 3.5, which shows the decreasing property of the SG estimators, can also be extended to general neural networks as well as other general neural networks.

REFERENCES

[1] Z. Allen-Zhu, Y. Li, and Y. Liang, *Learning and generalization in overparameterized neural networks, going beyond two layers*, arXiv preprint arXiv:1811.04918, (2018).

[2] AnonymousAuthors, *The impact of the mini-batch size on the dynamics of sgd: Variance and beyond (full version)*, (2024), https://github.com/AnonymousAuthorsXYZ/SGD-Variance.

[3] L. Bottou, *Stochastic gradient learning in neural networks*, Proceedings of Neuro-Nimes, 91 (1991), p. 12.

[4] L. Bottou, *Online learning and stochastic approximations*, On-line Learning in Neural Networks, 17 (1998), p. 142.

[5] L. Bottou, F. E. Curtis, and J. Nocedal, *Optimization methods for large-scale machine learning*, SIAM Review, 60 (2018), pp. 223–311.

[6] S. Boyd, S. P. Boyd, and L. Vandenberghe, *Convex optimization*, Cambridge university press, 2004.

[7] S. Du, X. Zhai, B. Poczos, and A. Singh, *Gradient descent provably optimizes overparameterized neural networks*, arXiv preprint arXiv:1810.02054, (2018).

[8] J. Fan, C. Ma, and Y. Zhong, *A selective overview of deep learning*, arXiv preprint arXiv:1904.05526, (2019).

[9] I. Goodfellow, Y. Bengio, and A. Courville, *Deep learning*, MIT press, 2016.

[10] E. Gorbunov, F. Hanzely, and P. Richtárik, *A unified theory of sgd: Variance reduction, sampling, quantization and coordinate descent*, in International Conference on Artificial Intelligence and Statistics, 2020, pp. 680–690.

[11] R. M. Gower, N. Loizou, X. Qian, A. Sailanbayev, E. Shulgin, and P. Richtárik, *SGD: General analysis and improved rates*, in International Conference on Machine Learning, 2019, pp. 5200–5209.

[12] P. Goyal, P. Dollár, R. Girshick, P. Noordhuis, L. Wesolowski, A. Kyrola, A. Tulloch, Y. Jia, and K. He, *Accurate, large minibatch SGD: Training Imagenet in 1 hour*, arXiv preprint arXiv:1706.02677, (2017).

[13] K. He, X. Zhang, S. Ren, and J. Sun, *Deep residual learning for image recognition*, in Proceedings of the IEEE conference on Computer Vision and Pattern Recognition, 2016, pp. 770–778.

[14] G. Hinton, O. Vinyals, and J. Dean, *Distilling the knowledge in a neural network*, arXiv preprint arXiv:1503.02531, (2015).

[15] S. Hochreiter and J. Schmidhuber, *Flat minima*, Neural Computation, 9 (1997), pp. 1–42.

[16] E. Hoffer, I. Hubara, and D. Soudry, *Train longer, generalize better: closing the generalization gap in large batch training of neural networks*, in Advances in Neural Information Processing Systems, 2017, pp. 1731–1741.

[17] S. Jastrzebski, Z. Kenton, D. Arpit, N. Ballas, A. Fischer, Y. Bengio, and A. Storkey, *Three factors influencing minima in SGD*, arXiv preprint arXiv:1711.04623, (2017).

[18] R. Johnson and T. Zhang, *Accelerating stochastic gradient descent using predictive variance reduction*, in Advances in Neural Information Processing Systems, 2013, pp. 315–323.

[19] N. S. Keskar, J. Nocedal, P. T. P. Tang, D. Mudigere, and M. Smelyanskiy, *On large-batch training for deep learning: Generalization gap and sharp minima*, in 5th International Conference on Learning Representations, 2017, 2017.

[20] A. Khaled and P. Richtárik, *Better theory for sgd in the nonconvex world*, arXiv preprint arXiv:2002.03329, (2020).

[21] Y. LeCun, L. Bottou, Y. Bengio, and P. Haffner, *Gradient-based learning applied to document recognition*, Proceedings of the Institute of Electrical and Electronics Engineers, 86 (1998), pp. 2278–2324.

[22] Y. LeCun, L. Bottou, G. B. Orr, and K.-R. Müller, *Efficient backprop*, in Neural networks: Tricks of the trade, Springer, 2012, pp. 9–48.

[23] L. Lei, C. Ju, J. Chen, and M. I. Jordan, *Non-convex finite-sum optimization via SCSG methods*, in Advances in Neural Information Processing Systems, 2017, pp. 2348–2358.

[24] M. Li, T. Zhang, Y. Chen, and A. J. Smola, *Efficient mini-batch training for stochastic optimization*, in Proceedings of the 20th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, 2014, pp. 661–670.

[25] Q. Li, C. Tai, and W. E, *Stochastic modified equations and adaptive stochastic gradient algorithms*, in Proceedings of the 34th International Conference on Machine Learning, PMLR, 2017, pp. 2101–2110.

[26] Q. Li, C. Tai, and E. Weinan, *Stochastic modified equations and adaptive stochastic gradient algorithms*, in International Conference on Machine Learning, PMLR, 2017, pp. 2101–2110.

[27] Y. Li and Y. Liang, *Learning overparameterized neural networks via stochastic gradient descent on structured data*, arXiv preprint arXiv:1808.01204, (2018).

[28] S. Mandt, M. D. Hoffman, and D. M. Blei, *Stochastic gradient descent as approximate bayesian inference*, The Journal of Machine Learning Research, 18 (2017), pp. 4873–4907.

[29] Q. Meng, Y. Wang, W. Chen, T. Wang, Z.-M. Ma, and T.-Y. Liu, *Generalization error bounds for optimization algorithms via stability*, arXiv preprint arXiv:1609.08397, (2016).

[30] W. Mou, L. Wang, X. Zhai, and K. Zheng, *Generalization bounds of SGLD for non-convex learning: Two theoretical viewpoints*, in Conference On Learning Theory, 2018, pp. 605–638.

[31] A. Neelakantan, L. Vilnis, Q. V. Le, I. Sutskever, L. Kaiser, K. Kurach, and J. Martens, *Adding gradient noise improves learning for very deep networks*, arXiv preprint arXiv:1511.06807, (2015).

[32] A. Paszke, S. Gross, F. Massa, A. Lerer, J. Bradbury, G. Chanan, T. Killeen, Z. Lin, N. Gimelshein, L. Antiga, et al., *Pytorch: An imperative style, high-performance deep learning library*, in Advances in Neural Information Processing Systems, 2019, pp. 8024–8035.

[33] K. Petersen, M. Pedersen, et al., *The matrix cookbook, vol. 7*, Technical University of Denmark, 15 (2008).

[34] N. L. Roux, M. Schmidt, and F. R. Bach, *A stochastic gradient method with an exponential convergence rate for finite training sets*, in Advances in Neural Information Processing Systems, 2012, pp. 2663–2671.

[35] M. Schmidt, N. Le Roux, and F. Bach, *Minimizing finite sums with the stochastic average gradient*, Mathematical Programming, 162 (2017), pp. 83–112.

[36] S. Shalev-Shwartz and T. Zhang, *Stochastic dual coordinate ascent methods for regularized loss minimization*, Journal of Machine Learning Research, 14 (2013), pp. 567–599.

[37] P. Y. Simard, D. Steinkraus, and J. C. Platt, *Best practices for convolutional neural networks applied to visual document analysis*, in Seventh International Conference on Document Analysis and Recognition, 2013, pp. 958–963.

[38] S. L. Smith and Q. V. Le, *A bayesian perspective on generalization and stochastic gradient descent*, arXiv preprint arXiv:1710.06451, (2017).

[39] R. Sun, *Optimization for deep learning: theory and algorithms*, arXiv preprint arXiv:1912.08957, (2019).

[40] R. Sun, *Optimization for deep learning: An overview*, Journal of the Operations Research Society of China, 8 (2020), pp. 249–294.

[41] Z. Yang, Z. Dai, Y. Yang, J. Carbonell, R. R. Salakhutdinov, and Q. V. Le, *Xlnet: Generalized autoregressive pretraining for language understanding*, in Advances in Neural Information Processing Systems, 2019, pp. 5754–5764.

[42] X. Zhang, J. Zhao, and Y. LeCun, *Character-level convolutional networks for text classification*, in Advances in Neural Information Processing Systems, 2015, pp. 649–657.

[43] Y. Zhang, P. Liang, and M. Charikar, *A hitting time analysis of stochastic gradient langevin dynamics*, in Conference on Learning Theory, 2017, pp. 1980–2022.