Proceedings of Machine Learning Research vol 98:1–26, 2019 30th International Conference on Algorithmic Learning Theory

Stochastic Nonconvex Optimization with Large Minibatches

Weiran Wang WEIRANW@AMAZON.COM
Amazon Alexa
101 Main St
Cambridge, MA 02142, USA

Nathan Srebro NATI@TTIC.EDU
Toyota Technological Institute at Chicago
6045 S Kenwood Ave
Chicago, IL 60637, USA

Editors: Aurélien Garivier and Satyen Kale

Abstract

We study stochastic optimization of nonconvex loss functions, which are typical objectives for training neural networks. We propose stochastic approximation algorithms which optimize a series of regularized, nonlinearized losses on large minibatches of samples, using only first-order gradient information. Our algorithms provably converge to an approximate critical point of the expected objective with faster rates than minibatch stochastic gradient descent, and facilitate better parallelization by allowing larger minibatches.

Keywords: stochastic nonconvex optimization, minibatch stochastic gradient descent, minibatch-prox

1. Introduction

Machine learning algorithms ultimately try to optimize the performance of models in the population. Consider the following stochastic optimization (generalized learning) problem (Vapnik, 2000; Shalev-Shwartz et al., 2009):

$$\min_{\mathbf{w}} \ \phi(\mathbf{w}) := \mathbb{E}_{\xi\sim D}\left[\ell(\mathbf{w},\xi)\right] \tag{1}$$

where our goal is to learn a predictor $\mathbf{w}$ given the instantaneous (loss) function $\ell(\mathbf{w},\xi)$ and i.i.d. samples $\xi_1,\xi_2,\ldots$ from some unknown data distribution D. In this work, we focus on losses $\ell(\mathbf{w},\xi)$ that are *nonconvex* functions of $\mathbf{w}$; a prevalent example of this setting is the training of deep neural networks, where $\mathbf{w}$ denotes the collection of trainable weights of a deep learning model, and $\ell(\mathbf{w},\xi)$ measures the loss of prediction (e.g., classification or regression) for the sample ξ using weights $\mathbf{w}$.

Despite efforts in introducing higher order optimization methods to this problem, stochastic gradient descent (SGD) with minibatches and its variants (Bottou, 1991; LeCun et al., 1998; Duchi et al., 2011; Zeiler, 2012; Kingma and Ba, 2015) remain by far the most popular methods for training deep neural networks, due to its simplicity and superior performance than the alternatives. In the vanilla version of minibatch SGD, we compute the averaged gradient over a small set of samples (called minibatch), e.g. using the backpropagation algorithm, and simply take a step in the negative direction. The use of minibatch (as opposed to a single sample for estimating the gradient) makes the training process more stable as it reduces the variance of gradient estimate. Moreover, to

process the same amount of samples, it takes smaller number of updates if a larger minibatch size is used, and the backpropagation procedure on a larger minibatch can utilize massive parallelization of linear algebra routines provided by advanced computational hardware (GPUs and clusters). As a result, using a larger minibatch in SGD can potentially significantly reduce the parallel training time.

Recently, there has been some empirical analysis of minibatch SGD for non-convex problems, focusing mostly on practical issues such as the correct scaling of stepsize (learning rate) and momentum with minibatch size (Goyal et al., 2017; Hoffer et al., 2017). A prominent observation is that, by properly setting the stepsize, minibatch SGD works well (converges to similarly good test set performance) for a wide range of minibatch sizes, while large minibatches facilitate better parallelization. But intriguingly, beyond certain threshold of minibatch size, training result start to deteriorate and simply scaling the stepsize does not help. The primary goal of this work is to theoretically investigate the issue of minibatch size in stochastic nonconvex optimization, and to provide practical algorithms/guidance to training deep neural networks. Our analysis applies to smooth nonconvex instantaneous losses, and uses a characterization of nonconvex functions named almost-convexity, which we introduce below.

Problem setup In this work, we assume that the differentiable instantaneous loss $\ell(\mathbf{w}, \xi)$ is β-smooth and σ-almost convex. Recall that a function $f(\mathbf{w})$ is β-smooth if $\|\nabla f(\mathbf{w}) - \nabla f(\mathbf{w}')\| \leq \beta \|\mathbf{w} - \mathbf{w}'\|$ for all $\mathbf{w}, \mathbf{w}'$, and in this case we have the following quadratic approximation for $f(\mathbf{w})$:

$$\left| f(\mathbf{w}) - f(\mathbf{w}') - \langle \nabla f(\mathbf{w}'), \mathbf{w} - \mathbf{w}' \rangle \right| \leq \frac{\beta}{2} \left\| \mathbf{w} - \mathbf{w}' \right\|^2, \qquad \forall \mathbf{w}, \mathbf{w}'.$$

On the other hand, a nonconvex function $f(\mathbf{w})$ is σ-almost convex for $\sigma \geq 0$ if

$$f(\mathbf{w}) - f(\mathbf{w}') - \langle \nabla f(\mathbf{w}'), \mathbf{w} - \mathbf{w}' \rangle \geq -\frac{\sigma}{2} \left\| \mathbf{w} - \mathbf{w}' \right\|^2, \qquad \forall \mathbf{w}, \mathbf{w}'.$$

A convex function is 0-almost convex, and a β-smooth function is σ-almost convex for some $\sigma \leq \beta$. For a twice differentiable function that is both β-smooth and σ-almost convex, the eigenvalues of its Hessian matrix lie in $[-\sigma, \beta]$.

We now discuss some properties we need when accessing the stochastic objective through samples. First, the stochastic gradient estimated on a single sample is unbiased, i.e.,

$$\mathbb{E}\left[\nabla \ell(\mathbf{w}, \xi)\right] = \nabla \phi(\mathbf{w}), \qquad \forall \mathbf{w}.$$

Second, as is common in the stochastic optimization literature (see, e.g., Lan, 2012; Ghadimi and Lan, 2016), we assume and that the variance of the stochastic gradient is bounded by V^2, i.e.,

$$\mathbb{E}_\xi \left\| \nabla \ell(\mathbf{w}, \xi) - \nabla \phi(\mathbf{w}) \right\|^2 \leq V^2, \qquad \forall \mathbf{w}.$$

Denote by $\phi^* = \min_{\mathbf{w}} \phi(\mathbf{w})$ the (globally) minimum value of $\phi(\mathbf{w})$, which we assume to be finite. Since in general we can not hope to efficiently obtain the global minimum of a nonconvex objective, the reasonable goal here is to find an approximate critical point $\mathbf{w}$ satisfying for some $\varepsilon > 0$ that

$$\mathbb{E} \left\| \nabla \phi(\mathbf{w}) \right\|^2 \leq \varepsilon^2. \tag{2}$$

We are interested in the number of samples and the amount of computation needed to achieve this goal.

Significance of almost convexity One may wonder whether it is reasonable to assume the nonconvex objective to be almost convex. We note that, almost convexity arises from the optimization of general smooth nonconvex objectives. Based on the Hessian Lipschitz assumption, Carmon et al. (2017) have shown that one can alternate over the negative curvature descent algorithm (which eventually leads us to a point at which the Hessian has small negative eigenvalues) and optimizing almost-convex problems, to obtain overall faster convergence than gradient descent in the non-stochastic setting. In fact, the almost-convex procedure is shown to be the key to the faster rate; see their Section 4 and also Allen-Zhu (2017, Appendix A). These results motivate us to study the stochastic version of the almost-convex problems, under common assumptions used to analyze stochastic gradient descent. We verify that improvement obtained in the non-stochastic case does carry over to the stochastic case, and in turn facilitates better parallelism. On the other hand, while we assume above that each individual loss is almost convex, relaxation to the almost convexity of only the population objective will be discussed later.

1.1. Minibatch SGD for nonconvex stochastic optimization

The theoretical performance of minibatch SGD has been relatively well studied for convex objectives (Lan, 2012; Dekel et al., 2012; Cotter et al., 2011). After T minibatch gradient updates, each using a stochastic gradient estimated on b samples, the accelerated minibatch SGD algorithm on a convex $\phi(\mathbf{w})$ returns an iterate $\mathbf{w}$ satisfying

$$\mathbb{E}\left[\phi(\mathbf{w}) - \phi(\mathbf{w}_*)\right] \leq \mathcal{O}\left(\frac{\beta \left\|\mathbf{w}_0 - \mathbf{w}_*\right\|^2}{T^2} + \frac{V \left\|\mathbf{w}_0 - \mathbf{w}_*\right\|}{\sqrt{bT}}\right), \tag{3}$$

where $\mathbf{w}_0$ is the initialization, and $\mathbf{w}_* = \arg\min_{\mathbf{w}} \phi(\mathbf{w})$.

For stochastic nonconvex optimization, Ghadimi et al. (2016) analyzed the convergence of minibatch SGD under the same problem setup as ours, and Ghadimi and Lan (2016) further proposed a randomized stochastic accelerated gradient (RSAG) method which resembles the accelerated stochastic approximation method for convex optimization (Lan, 2012). After T minibatch gradient updates, each using stochastic gradient estimated on b samples, their algorithms return an iterate $\mathbf{w}$ satisfying[1]

$$\mathbb{E}\left\|\nabla\phi(\mathbf{w})\right\|^2 \leq \mathcal{O}\left(\frac{\beta(\phi(\mathbf{w}_0) - \phi^*)}{T} + \frac{V\sqrt{\beta(\phi(\mathbf{w}_0) - \phi^*)}}{\sqrt{bT}}\right). \tag{4}$$

To parse this result, we observe the following:

- When $V = 0$, or in other words exact gradients are used, the second term vanishes and the convergence rate reduces to $\|\nabla\phi(\mathbf{w})\|^2 \leq \mathcal{O}\left(\frac{\beta(\phi(\mathbf{w}_0)-\phi^*)}{T}\right)$, recovering the rate for deterministic gradient descent (Ghadimi et al., 2016). We refer to this term as the "optimization error" since it is independent of the samples.
- The second term in (4) results from the noise in stochastic gradients, and we refer to it as the "sample error" since it results from the sampling process. This term is asymptotically

1. This can be deduced from eqn (3.20) of Ghadimi and Lan (2016), as the variance of stochastic gradient reduces to V^2/b when b samples are used in estimating the gradient.

dominant as long as $b = \mathcal{O}\left(\frac{V^2 T}{\beta(\phi(\mathbf{w}_0)-\phi^*)}\right)$. Using a much larger b, the first term of (4) becomes dominant but since the first term is independent of b, the algorithm is no longer sample efficient (it is using more fresh samples than needed). This is consistent with the empirical findings of practitioners of minibatch SGD: beyond certain minibatch size, learning slows down and in particular, the objective on test set (which is an estimate of the population objective) do not decrease faster with the amount of computation, even though more samples and computation (e.g., backpropagation) is involved in each stochastic gradient update.

Denote the total number of samples used by $N = bT$. The convergence rate (4) indicates that, to find an critical point satisfying (2), the total sample needed is $N(\varepsilon) = \mathcal{O}\left(\frac{V^2\beta(\phi(\mathbf{w}_0)-\phi^*)}{\varepsilon^4}\right)$, while the maximum minibatch size that maintains sample efficiency, and the iteration complexity using this minibatch size, are respectively

$$b_{\mathrm{RSAG}} = \mathcal{O}\left(\frac{V\sqrt{N(\varepsilon)}}{\sqrt{\beta(\phi(\mathbf{w}_0)-\phi^*)}}\right) = \mathcal{O}\left(\frac{V^2}{\varepsilon^2}\right), \qquad T_{\mathrm{RSAG}} = \mathcal{O}\left(\frac{\beta(\phi(\mathbf{w}_0)-\phi^*)}{\varepsilon^2}\right). \tag{5}$$

An optimal choice of minibatch size (up to constants) for this method is thus b_{RSAG}—in the regmime $b < b_{\mathrm{RSAG}}$, increasing the minibatch size leads to reduced number of iterations, down to a minimum of T_{RSAG}, but any further increase would increase the overall work performed (overall number of vector operations), without decreasing the required number of iterations.

In a highly parallel setting, the number of mini-batch gradient evaluations T captures the *parallel runtime*, since each such evaluation can be efficiently parallelized over the machines, and so throughout we account of the parallel runtime in terms of the number of such mini-batch gradient evaluations, regardless of the number of points involved in each such mini-batch. We also account for the overall work performed (or energy consumed), in terms of the overall number of gradient evaluations or vector operations. For mini-batch SGD we process each point once, and so the overall work is $\mathcal{O}(N(\varepsilon))$.

1.2. Iterative convexification methods for deterministic nonconvex optimization

The approach discussed in Section 1.1 draws fresh samples in each update. A alternative approach is to draw a single set of n training examples and minimize the empirical risk (or sample averaged approximation) on these points using deterministic optimization techniques:

$$\min_{\mathbf{w}} \ \hat{\phi}(\mathbf{w}) = \frac{1}{n}\sum_{i=1}^{n} \ell(\mathbf{w}, \xi_i). \tag{6}$$

We could attempt to optimize (6) using gradient descent. But recently, Carmon et al. (2017) and Allen-Zhu (2017) demonstrated that when $\sigma \ll \beta$, i.e., when the objective is not too nonconvex (the negative eigenvalues are not too large in magnitude), one can obtain faster convergence than gradient descent, by transforming an almost-convex objective into a series of (strongly) convex optimization problems which are then solved very efficiently by batch accelerated gradient descent; this technique will be discussed in detail in Section 2.1. Applying this technique to (6), to obtain a

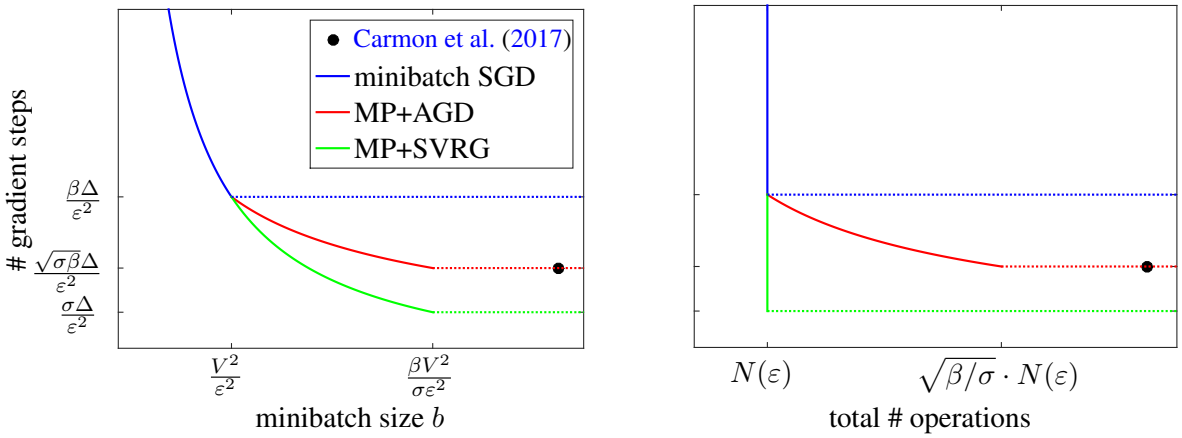

Figure 1: Illustration of theoretical guarantees for our algorithms (MP equipped with two convex optimizers—AGD and SVRG) and the comparisons with minibatch SGD, in terms of number of gradient steps vs. the minibatch size b (left plot), and number of gradient steps vs. the total number of vector operations and gradient calculations. Here $\Delta = \phi(\mathbf{w}_0) - \phi^*$. We have assumed for MP+SVRG that the parallel runtime for computing the gradient on a minibatch of b samples (with communications) is much larger than computing the gradient on a single example locally, see discussion in Section 3. The regimes that are not sample-efficient are dotted.

$\mathbf{w}$ satisfying $\left\|\nabla\hat{\phi}(\mathbf{w})\right\|^2 \leq \varepsilon^2$, it is sufficient to perform[2]

$$T_{\text{BATCH}} = \tilde{\mathcal{O}}\left(\frac{\sqrt{\sigma\beta}(\hat{\phi}(\mathbf{w}_0) - \min_{\mathbf{w}}\hat{\phi}(\mathbf{w}))}{\varepsilon^2}\right) \tag{7}$$

exact (batch) gradient-based updates. Since again each such batch computation can be calculated in parallel, we get a parallel runtime of $\mathcal{O}(T_{\text{BATCH}})$ and the total number of gradient evaluations required is $\mathcal{O}(T_{\text{BATCH}} \cdot N(\varepsilon))$.

We suggest algorithms that are based on the same intuition and use the same convexification procedure, but we propose tackling the stochastic optimization problem directly, using approximate gradients obtained from minibatches at each iteration. This has two advantages. First, we are tackling the stochastic optimization objective directly, which is many cases is our true objective, and so obtain guarantees directly on the population $\|\nabla\phi(\mathbf{w})\|$ rather than merely its empirical approximation $\left\|\nabla\hat{\phi}(\mathbf{w})\right\|$. Second, as we shall see, but using only part of the data at each iteration, instead of the entire data set, we can reduce the total amount of work performed (total number of gradient computations and vector operations) without sacrificing the parallel runtime and accuracy guarantees.

2. This amounts to plugging in $\gamma = \sigma$ in Carmon et al. (2017)[Lemma 3.1] for their `Almost-Convex-AGD` algorithm. Note that the authors also showed the $\tilde{\mathcal{O}}\left(\frac{1}{\varepsilon^{7/4}}\right)$ iteration complexity is achievable with second order information.

1.3. Our contributions

We propose stochastic approximation algorithms that provably converge to an approximate critical point of the nonconvex population objective. In our template algorithm, which we refer to as "minibatch-prox" (MP), we draw b fresh samples at each iteration and approximately optimize a convex objective defined by these samples:

$$\mathbf{w}_t \approx \arg\min_{\mathbf{w}} \frac{1}{b}\sum_{i=1}^{b} \ell(\mathbf{w}, \xi_i^t) + \frac{\gamma}{2}\left\|\mathbf{w} - \mathbf{w}_{t-1}\right\|^2 \qquad \text{where} \quad \gamma > \sigma, \quad \text{for} \quad t = 1, \ldots, K.$$

Choosing a random iterate $\mathbf{w}$ from the first K iterations of MP, we have that (for large enough b)

$$\mathbb{E}\left\|\nabla\phi(\mathbf{w})\right\|^2 \leq \mathcal{O}\left(\frac{\sigma(\phi(\mathbf{w}_0) - \phi^*)}{K} + \frac{V\sqrt{\beta(\phi(\mathbf{w}_0) - \phi^*)}}{\sqrt{bK}}\right). \tag{8}$$

MP can use larger minibatch while maintaining sample efficiency, at the cost of more complicated operations (solvings nonlinear optimization problems) on each minibatch: With a minibatch size of $\Theta\left(\frac{\beta V^2}{\sigma\varepsilon^2}\right)$, MP solves $\mathcal{O}\left(\frac{\sigma(\phi(\mathbf{w}_0)-\phi^*)}{\varepsilon^2}\right)$ convex subproblems.

MP is a meta algorithm and allows us to plug in different optimizers for solving the convex subproblems on each minibatch. In particular, when accelerated gradient descent is used as the optimizer, the total number of gradient steps in MP+AGD with a minibatch of size $\Theta\left(\frac{\beta V^2}{\sigma\varepsilon^2}\right)$ is

$$T_{\text{MP-AGD}} = \tilde{\mathcal{O}}\left(\frac{\sqrt{\sigma\beta}(\phi(\mathbf{w}_0) - \phi^*)}{\varepsilon^2}\right).$$

This significantly improves T_{RSAG} in (5) when $\beta \gg \sigma$, and this is achieved at the cost of a larger total computational cost (i.e., total number of vector operations and gradient computations). The comparison is depicted in Figure 1. In particular, the right panel of the Figure we see how RSAG and MP+AGD compare in terms of the parallel runtime (number of gradient steps) and total work (number of individual gradient computations). The Pareto optimal point for these two resources using RSAG is given by T_{RSAG} and $N(\varepsilon)$ respectively. MP+AGD does not dominate RSAG, but rather allows us reduce the parallel runtime at the cost of increasing the total computation cost, down to a minimum parallel runtime of $T_{\text{MP-AGD}}$. This is the same parallel runtime required by the batch methods, but we dominate them since to achieve this runtime, we still require significantly less total computation.

We also analyze the use of other convex optimizers in MP, and develop a memory-efficient version for it when the total number of samples used in each convex subproblem is very large.

Our results have important implications for parallel or distributed learning: they suggest that it is possible to significantly reduce parallel runtime, but that this requires going beyond the minibatch SGD paradigm of using each minibatch only once.

2. Minibatch-prox (MP) for nonconvex smooth loss

In this section, we first review the fundamental convexification step in our algorithms which allows us to find approximate critical point by solving convex subproblems, and then propose the basic version of our algorithm and analyze its convergence properties.

2.1. Convexification of nonconvex problems

The key ingredient in our algorithms is the reduction from the optimization of a nonconvex objective into the optimization of a series of convex problems (Bertsekas, 1979, 1999; Carmon et al., 2017; Allen-Zhu, 2017). Consider the following iterative procedure: for $t = 1, \ldots, K$,

$$\mathbf{w}_t \approx \mathbf{w}_t^* = \arg\min_{\mathbf{w}} \; F_t(\mathbf{w}) \quad \textit{where} \quad F_t(\mathbf{w}) := \phi(\mathbf{w}) + \frac{\gamma}{2} \|\mathbf{w} - \mathbf{w}_{t-1}\|^2 \tag{9}$$

where $\gamma > \sigma$. At each iteration of this algorithm, one approximately minimizes a regularized objective, where the ℓ_2 proximity term encourages the new iterate to be close to the previous iterate. With the regularization term, the objective $F_t(\mathbf{w})$ is $(\gamma - \sigma)$-strongly convex and the global minimizer $\mathbf{w}_t^*$ is unique. Similar procedures have also been used when $\phi(\mathbf{w})$ is convex to speedup first order methods (Lin et al., 2015).

We can quantify the number of convex subproblems to be solved in (9), so as to find an approximate critical point of $\phi(\mathbf{w})$. Assume for now that we always obtain the exact minimizer of the subproblem (9) at each iteration t, i.e., $\mathbf{w}_t = \mathbf{w}_t^*$. Then we have by the first order optimality of $\mathbf{w}_t^*$ that $\nabla\phi(\mathbf{w}_t) = \gamma(\mathbf{w}_{t-1} - \mathbf{w}_t)$. And in view of this optimality condition, $\mathbf{w}_t$ is an approximate critical point if $\|\mathbf{w}_{t-1} - \mathbf{w}_t\| \le \varepsilon/\gamma$. On the other hand, if $\|\mathbf{w}_{t-1} - \mathbf{w}_t\| > \varepsilon/\gamma$, we expect to achieve large reduction in $\phi(\mathbf{w})$: owing to the strong convexity of $F_t(\mathbf{w})$, we have $F(\mathbf{w}_{t-1}) - F(\mathbf{w}_t) \ge \frac{\gamma-\sigma}{2} \|\mathbf{w}_{t-1} - \mathbf{w}_t\|^2$, which is equivalent to

$$\phi(\mathbf{w}_{t-1}) - \phi(\mathbf{w}_t) \ge \frac{2\gamma - \sigma}{2} \|\mathbf{w}_{t-1} - \mathbf{w}_t\|^2 > \frac{\varepsilon^2}{2\gamma}.$$

We can not keep decreasing the objective in this way for more than $\frac{2\gamma(\phi(\mathbf{w}_0)-\phi^*)}{\varepsilon^2}$ iterations.

The following simple lemma makes this intuition more precise when we have an approximate minimizer at each iteration. A similar and more general result (which applies to constrained optimization/composite objectives) can be found in Allen-Zhu (2017, Lemma 4.1).

Lemma 1 *Let $F_t(\mathbf{w})$ be defined in (9). If we apply a possibly randomized algorithm $\mathcal{A}$ to obtain an approximate minimizer $\mathbf{w}_t$ satisfying*

$$\mathbb{E}_{\mathcal{A}} \left[F_t(\mathbf{w}_t) - F_t(\mathbf{w}_t^*) \,|\, \mathbf{w}_{t-1} \right] \le \epsilon, \tag{10}$$

then we have

$$\mathbb{E}_{\mathcal{A}} \|\nabla\phi(\mathbf{w}_t)\|^2 \le 2\gamma^2 \mathbb{E}_{\mathcal{A}} \|\mathbf{w}_{t-1} - \mathbf{w}_t\|^2 + 4(\beta + \gamma)\epsilon. \tag{11}$$

Corollary 2 *If we run the iterative procedure (9) for K iterations, optimizing each $F_t(\mathbf{w})$ to ϵ-suboptimality with an algorithm $\mathcal{A}$, i.e., $\mathbb{E}_{\mathcal{A}} \left[F_t(\mathbf{w}_t) - F_t(\mathbf{w}_t^*) \,|\, \mathbf{w}_{t-1} \right] \le \epsilon$ for $t = 1, \ldots, K$, and pick an iteration index $R \in \{1, \ldots, K\}$ uniformly at random, we have*

$$\mathbb{E}_{R,\mathcal{A}} \|\nabla\phi(\mathbf{w}_R)\|^2 \le \frac{4\gamma \left(\phi(\mathbf{w}_0) - \phi^*\right)}{K} + (4\beta + 8\gamma)\epsilon.$$

2.2. The basic MP algorithm

We now turn to solving the subproblems in (9), i.e., to minimizing the *stochastic* convex objective $F_t(\mathbf{w})$, to which we only have access through i.i.d. samples of the underlying distribution. A natural approach for this problem is through (approximate) empirical risk minimization: we draw b i.i.d. samples $Z_t = \{\xi_1^t, \ldots, \xi_b^t\}$ and compute

$$\text{(MP)} \quad \mathbf{w}_t \approx \hat{\mathbf{w}}_t = \arg\min_{\mathbf{w}} \ \hat{F}_t(\mathbf{w}) \quad \text{where } \hat{F}_t(\mathbf{w}) := \frac{1}{b}\sum_{i=1}^{b} \ell(\mathbf{w}, \xi_i^t) + \frac{\gamma}{2}\left\|\mathbf{w} - \mathbf{w}_{t-1}\right\|^2. \quad (12)$$

Since this procedure optimizes on a minibatch of samples the nonlinearized loss (as opposed to linearized loss which leads to minibatch SGD), we call this procedure "minibatch-prox" (MP). Similar algorithms were proposed previously for optimizing convex problems: a version of this algorithm with $b = 1$ was studied under the names "passive aggressive" update (Crammer et al., 2006) and "implicit gradient descent" (Kulis and Bartlett, 2010) for online learning, was used for optimizing finite-sum objectives (Bertsekas, 2015; Defazio, 2016), and the version of large b was more recently used as the building block to develop communication-efficient distributed algorithms for stochastic convex optimization (Li et al., 2014; Wang et al., 2017).

In order to ensure small suboptimality in the population objective $F_t(\mathbf{w})$ as required by Corollary 2, we need to bound the difference between the empirical and the population objectives at $\mathbf{w}_t$, or in other words, the generalization performance of $\mathbf{w}_t$. In the following lemma, we provide the generalization guarantee based on the notion of stability (Bousquet and Elisseeff, 2002; Shalev-Shwartz et al., 2009). Our result establishes the connection between the stability of ERM and the variance of stochastic gradients, which have been two major and seemingly parallel assumptions for deriving stochastic learning guarantees, and this formal connection appears to be new in the literature.

Lemma 3 *Consider the stochastic optimization problem*

$$F(\mathbf{w}) := \phi(\mathbf{w}) + r(\mathbf{w}) = \mathbb{E}_\xi\left[\ell(\mathbf{w}, \xi)\right] + r(\mathbf{w})$$

where the instantaneous loss $\ell(\mathbf{w}, \xi)$ is σ-almost convex and β-smooth in $\mathbf{w}$, and satisfies the variance condition $\mathbb{E}_\xi \left\|\nabla\ell(\mathbf{w}, \xi) - \nabla\phi(\mathbf{w})\right\|^2 \le V^2$, and the data-independent regularizer $r(\mathbf{w})$ is γ-strongly convex with $\gamma > \sigma$. Denote $\mathbf{w}^* = \arg\min_{\mathbf{w}} \ F(\mathbf{w})$.

Let $Z = \{\xi_1, \ldots, \xi_b\}$ be i.i.d. samples and

$$\hat{\mathbf{w}} = \arg\min_{\mathbf{w}} \ \hat{F}(\mathbf{w}) \qquad \textit{where} \quad \hat{F}(\mathbf{w}) := \hat{\phi}(\mathbf{w}) + r(\mathbf{w}) = \frac{1}{b}\sum_{i=1}^{b} \ell(\mathbf{w}, \xi_i) + r(\mathbf{w}).$$

Assume that $(\gamma - \sigma)b \ge 2(\sigma + \beta)$. Then the following stability results hold.

1. *For the regularized empirical risk minimizer $\hat{\mathbf{w}}$, we have*

$$\mathbb{E}_Z\left[F(\hat{\mathbf{w}}) - F(\mathbf{w}^*)\right] \le \mathbb{E}_Z\left[\phi(\hat{\mathbf{w}}) - \hat{\phi}(\hat{\mathbf{w}})\right] \le \frac{8V^2}{(\gamma - \sigma)b}.$$

2. *If a possibly randomized algorithm $\mathcal{A}$ minimizes $\hat{F}(\mathbf{w})$ up to δ-suboptimality, i.e., $\mathcal{A}$ returns an approximate solution $\tilde{\mathbf{w}}$ such that*

$$\mathbb{E}_{Z,\mathcal{A}}\left[\hat{F}(\tilde{\mathbf{w}}) - \hat{F}(\hat{\mathbf{w}})\right] \leq \delta,$$

we have

$$\mathbb{E}_{Z,\mathcal{A}}\left[F(\tilde{\mathbf{w}}) - F(\hat{\mathbf{w}})\right] \leq \frac{8V^2}{(\gamma-\sigma)b} + \frac{2(\beta+\gamma)\delta}{\gamma-\sigma},$$
$$\mathbb{E}_{Z,\mathcal{A}}\left[F(\tilde{\mathbf{w}}) - F(\mathbf{w}^*)\right] \leq \frac{16V^2}{(\gamma-\sigma)b} + \frac{2(\beta+\gamma)\delta}{\gamma-\sigma}.$$

The above lemma shows that, the convergence rate of the stochastic objective $F_t(\mathbf{w})$ by ERM is of the order $\mathcal{O}\left(\frac{1}{b}\right)$, and thus in order to achieve ϵ-suboptimality in $F_t(\mathbf{w})$, it suffices to exactly solve the ERM problem defined by $\mathcal{O}\left(\frac{1}{\epsilon}\right)$ samples. Moreover, the second part of Lemma 3 shows that as long as we minimize the ERM objective $\hat{F}_t(\mathbf{w})$ to suboptimality $\delta = \mathcal{O}(\epsilon)$, the population suboptimality remains of the order $\mathcal{O}(\epsilon)$. Allowing inexact minimization enables us to use state-of-the-art methods for convex optimization.

Remark 4 (Relaxation of individual almost convexity) *From the proof of Lemma 3, we observe that the most important usage of the almost convexity is to ensure that the regularized empirical objective $\hat{F}(\mathbf{w})$ is $(\gamma-\sigma)$-strongly convex; as long as this holds, we obtain the $\Omega\left(\frac{V^2}{(\gamma-\sigma)b}\right)$ stability for $b \geq \frac{4\beta}{\gamma-\sigma}$, without almost convexity of the instantaneous losse. As a result, we may relax our assumption to the population loss $\phi(\mathbf{w})$ being σ-almost convex. Based on the β-smoothness of $\ell(\mathbf{w},\xi)$ and matrix concentration, with high probability it holds that $\hat{\phi}(\mathbf{w})$ is $\Omega(\sigma)$-almost convex as long as $b \succeq \frac{\beta^2 \log d}{\sigma^2}$, where d is the dimensionality of $\mathbf{w}$. We will see shortly that our optimal minibatch size shall increase with the final accuracy, so as $\varepsilon \to 0$, we indeed have that all subproblems are sufficiently strongly convex and consequently, the same results hold with high probability.*

2.3. Convergence of the basic MP algorithm

We are now ready to analyze the convergence property of MP.

Theorem 5 *Set $\gamma = \sigma + \sqrt{\frac{32(\beta+2\sigma)V^2K}{(\phi(\mathbf{w}_0)-\phi^*)b}}$ in the minibatch-prox algorithm (12). And assume that for each iteration t, we draw b samples to approximate $F_t(\mathbf{w})$ with $b \geq \frac{2(\sigma+\beta)}{\gamma-\sigma}$, and minimize the ERM objective $\hat{F}_t(\mathbf{w})$ using a randomized algorithm $\mathcal{A}$, such that $\mathbb{E}_{Z_t,\mathcal{A}}\left[\hat{F}_t(\mathbf{w}_t) - \hat{F}_t(\hat{\mathbf{w}}_t) \,|\, \mathbf{w}_{t-1}\right] \leq \delta = \frac{8V^2}{(\beta+\gamma)b}$. If we pick an iteration index $R \in \{1,\ldots,K\}$ uniformly at random, we have*

$$\mathbb{E}_{R,\mathcal{A}}\left\|\nabla\phi(\mathbf{w}_R)\right\|^2 \leq \frac{4\sigma(\phi(\mathbf{w}_0)-\phi^*)}{K} + \frac{256V^2}{b} + \frac{32V\sqrt{(2\beta+4\sigma)(\phi(\mathbf{w}_0)-\phi^*)}}{\sqrt{bK}}$$

We now add a few remarks regarding the convergence result in Theorem 5. First, the algorithm does not converge to an approximate critical point for very small b, and in fact to satisfy (2) it is

necessary to have $b \succeq \frac{V^2}{\varepsilon^2}$. But once $b \succeq \frac{V^2 K}{\beta(\phi(\mathbf{w}_0)-\phi^*)}$, the second term $\frac{256V^2}{b}$ is dominated by the other two terms, and we obtain the convergence guarantee

$$\mathbb{E}_{R,\mathcal{A}} \left\|\nabla\phi(\mathbf{w}_R)\right\|^2 \leq \mathcal{O}\left(\frac{\sigma(\phi(\mathbf{w}_0)-\phi^*)}{K} + \frac{V\sqrt{\beta(\phi(\mathbf{w}_0)-\phi^*)}}{\sqrt{bK}}\right).$$

Compare this with the convergence rate of RSAG given in (4). We note that, while the second term ("sample error") is of the same order for both methods, the first term ("optimization error") in our method depends on σ instead of β. The first term also agrees with the number of subproblems resulted from the convexification procedure (cf. the discussion in Section 2.1).

Further assume that the second term is dominant, which is true as long as $b = \mathcal{O}\left(\frac{\beta V^2 K}{\sigma^2(\phi(\mathbf{w}_0)-\phi^*)}\right)$, then to find an critical point satisfying (2), the sample complexity is $N(\varepsilon) = \mathcal{O}\left(\frac{V^2\beta(\phi(\mathbf{w}_0)-\phi^*)}{\varepsilon^4}\right)$, while the maximum minibatch size (that maintains sample efficiency) and the iteration complexity using this minibatch size are respectively

$$b_{\mathrm{MP}} = \mathcal{O}\left(\frac{\sqrt{\beta}V\sqrt{N(\varepsilon)}}{\sqrt{\sigma^2(\phi(\mathbf{w}_0)-\phi^*)}}\right) = \mathcal{O}\left(\frac{\beta V^2}{\sigma\varepsilon^2}\right), \qquad K_{\mathrm{MP}} = \mathcal{O}\left(\frac{\sigma(\phi(\mathbf{w}_0)-\phi^*)}{\varepsilon^2}\right). \tag{13}$$

Increasing the minibatch size reduces the number of iterations in the regime of $b \leq b_{\mathrm{MP}}$. Therefore, MP achieves the same sample error as RSAG using the same level of samples, but when $\sigma \ll \beta$, MP allows us to use much larger minibatch size and a smaller number of minibatches. The caveat here, which we will address in the next section, is that MP requires solving an optimization problem on each minibatch, as opposed to performing a single gradient step.

Stepsize By the optimality condition of (12), we have $\mathbf{w}_t \approx \mathbf{w}_{t-1} - \frac{1}{\gamma}\left(\frac{1}{b}\sum_{i=1}^{b}\nabla\ell(\mathbf{w}_t,\xi_i^t)\right)$. This update resembles that of minibatch SGD, except that the gradient is evaluated at the "future" iterate. Moreover, according to Theorem 5, the "stepsize" $\frac{1}{\gamma}$ roughly varies like $\sqrt{\frac{b}{K}}$ (this approximation is more accurate for smaller b, in which case $\sigma \ll \gamma$), which scales with b if the number of total samples N is fixed, and scales with $\sqrt{b}$ if the number of iterations K is fixed, consistent with the findings of Goyal et al. (2017) and Hoffer et al. (2017) respectively for minibatch SGD.

Remark 6 *We have shown that it suffices to approximately minimize the stochastic objective $F_t(\mathbf{w})$ with $\mathcal{O}\left(\frac{1}{\varepsilon^2}\right)$ samples, by approximately minimizing the empirical objective $\hat{F}_t(\mathbf{w})$. But the number of samples can be too large (as $\varepsilon \to 0$) that the memory requirement is high, since we need to store this many samples and process them multiple times. In Appendix B, we provide a modified algorithm to resolve this issue, which achieves the same learning guarantee with the same level of total samples, and using any (sufficiently large) minibatch size.*

We demonstrate our theory and the MP algorithm on deep neural networks training in Appendix A.

3. Distributed implementation of MP

In the previous Section, we presented the template algorithm MP. But MP requires solving an optimization problem on a mini-batch at each iteration. We now instantiate the algorithm by suggesting

Table 1: Comparisons between minibatch SGD and MP equipped with two convex optimizers, in terms of both the runtime and the total number of vector operations to find an approximate critical point. The runtime is measured by the number of batch gradient evaluations, each of which cost τ_b on a minibatch of b samples, and the number of serial gradient descent updates, each of which cost τ_1 on a single sample. Denote by $\Delta := \phi(\mathbf{w}_0) - \phi^*$ the initial suboptimality. The (sufficient) sample complexity is denoted by $N(\varepsilon) = \mathcal{O}\left(\frac{V^2\beta\Delta}{\varepsilon^4}\right)$. We hide poly-logarithmic dependence on $(\sigma, \beta, V, \Delta, \varepsilon, b)$. Regimes that are not sample-efficient are shadowed.

	Minibatch size	Parallel runtime	# vector operations
RSAG	$b \prec \frac{V^2}{\varepsilon^2}$	$\frac{N(\varepsilon)}{b} \times \tau_b$	$N(\varepsilon)$
	$b \asymp \frac{V^2}{\varepsilon^2}$	$\frac{\beta\Delta}{\varepsilon^2} \times \tau_b$	$N(\varepsilon)$
	$b \succ \frac{V^2}{\varepsilon^2}$	$\frac{\beta\Delta}{\varepsilon^2} \times \tau_b$	$\frac{b\varepsilon^2}{V^2} \cdot N(\varepsilon)$
MP + AGD	$\frac{V^2}{\varepsilon^2} \preceq b \prec \frac{\beta V^2}{\sigma\varepsilon^2}$	$\sqrt{\frac{\varepsilon^2}{V^2 b}} \cdot N(\varepsilon) \times \tau_b$	$\sqrt{\frac{b\varepsilon^2}{V^2}} \cdot N(\varepsilon)$
	$b \asymp \frac{\beta V^2}{\sigma\varepsilon^2}$	$\frac{\sqrt{\sigma\beta}\Delta}{\varepsilon^2} \times \tau_b$	$\sqrt{\frac{\beta}{\sigma}} \cdot N(\varepsilon)$
	$b \succ \frac{\beta V^2}{\sigma\varepsilon^2}$	$\left(\frac{b\varepsilon^2}{V^2}\right)^{\frac{1}{4}} \cdot \frac{\sigma^{\frac{3}{4}}\beta^{\frac{1}{4}}\Delta}{\varepsilon^2} \times \tau_b$	$\left(\frac{b\varepsilon^2}{V^2}\right)^{\frac{5}{4}} \left(\frac{\sigma}{\beta}\right)^{\frac{3}{4}} \cdot N(\varepsilon)$
MP + SVRG	$\frac{V^2}{\varepsilon^2} \preceq b \prec \frac{\beta V^2}{\sigma\varepsilon^2}$	$\frac{N(\varepsilon)}{b} \times \tau_b + \frac{\beta\Delta}{\varepsilon^2} \times \tau_1$	$N(\varepsilon)$
	$b \asymp \frac{\beta V^2}{\sigma\varepsilon^2}$	$\frac{\sigma\Delta}{\varepsilon^2} \times \tau_b + \frac{\beta\Delta}{\varepsilon^2} \times \tau_1$	$N(\varepsilon)$
	$b \succ \frac{\beta V^2}{\sigma\varepsilon^2}$	$\frac{\sigma\Delta}{\varepsilon^2} \times \tau_b + \sqrt{\frac{b\varepsilon^2}{V^2} \cdot \frac{\sqrt{\sigma\beta}\Delta}{\varepsilon^2}} \times \tau_1$	$\frac{b\varepsilon^2}{V^2} \cdot \frac{\sigma}{\beta} \cdot N(\varepsilon)$

specific distributed procedures for solving this minibatch optimization problem. We consider two possible solvers for the convex subproblems $\hat{F}_t(\mathbf{w}), t = 1, \ldots, K$ in MP, and discuss the resulting overall parallel runtime and the total computational cost.

Accelerated gradient descent The first choice is the (distributed) accelerated gradient descent (AGD, Nesterov, 2004), in which case MP uses the same minibatch gradients as minibatch SGD does. Observe that each $\hat{F}_t(\mathbf{w})$ is both $(\beta + \gamma)$-smooth and $(\gamma - \sigma)$-strongly convex, and by our choice of γ, its condition number is $\kappa = \frac{\beta+\gamma}{\gamma-\sigma} = \mathcal{O}\left(\sqrt{\frac{\beta(\phi(\mathbf{w}_0)-\phi^*)b}{V^2 K}}\right)$ which increases with b. By the convergence rate of AGD, when minimizing $\hat{F}_t(\mathbf{w})$, the number of gradient descent updates needed to achieve δ suboptimality is $\mathcal{O}\left(\sqrt{\kappa}\log\frac{1}{\delta}\right)$. Consequently, the total number of gradient descent updates throughout the MP algorithm is[3] $\tilde{\mathcal{O}}\left(K \cdot \sqrt{\kappa}\right)$ and the total number of vector operations by the algorithm is $\tilde{\mathcal{O}}\left(b \cdot K \cdot \sqrt{\kappa}\right)$. We provide the total number of gradient steps and the corresponding computational cost, as functions of the problem parameters, for different regimes of minibatch size in Table 1. Most notably, when using the maximum minibatch size b_{MP}, the total number of gradient descent updates is

$$T_{\text{MP-AGD}} = \tilde{\mathcal{O}}\left(\frac{\sqrt{\sigma\beta}(\phi(\mathbf{w}_0) - \phi^*)}{\varepsilon^2}\right), \tag{14}$$

3. We use the $\tilde{\mathcal{O}}(\cdot)$ notation to hide poly-logarithmic dependence on $(\sigma, \beta, V, \phi(\mathbf{w}_0) - \phi^*, \varepsilon, b)$.

which is asymptotically smaller than T_{RSAG}. In the *worst scenario* where $\sigma = \beta$, we perform roughly the same number of gradient descent updates as RSAG. But when $\sigma \ll \beta$, we can significantly reduce the parallel runtime.

Distributed SVRG Another option for optimizing $\hat{F}_t(\mathbf{w})$ is the distributed SVRG algorithm for finite-sum problems (Lee et al., 2016; Shamir, 2016).[4] This algorithm alternates over two types of operations: the evaluation of batch gradient on b samples, which can distributed into multiple machines, and many serial stochastic gradient descent steps, each of which uses gradient computed on a single sample on one local machine.

Denote the time cost for the two types of access by τ_b and τ_1 respectively. According to the convergence of distributed SVRG, the runtime needed to optimize each subproblem to sufficient accuracy is

$$\tilde{\mathcal{O}}\left(\tau_b + \kappa\tau_1\right).$$

Similar to the case of AGD, we provide the runtime and the total computational cost by MP+SVRG in Table 1.

The parallel runtime of the two types of operations, calculating the gradient of a mini-batch of size b in parallel across machines, and calculating the gradient on a single sample locally, are not directly relate-able. From a pure parallel computation perspective, one could argue that τ_b does not depend much on b, since we can distribute the computation across b machines, and so $\tau_1 \approx \tau_b$. If this is the case, MP+SVRG does not provide any advantage over RSAG, since MP+SVRG's parallel runtime will be dominates by $\frac{\beta\Delta}{\epsilon^2}$, the same as RSAG's.

However, more realistically, even in a parallel setting, we would expect computing the gradient of a single point on a single machine, without any communication, would be much quicker than parallel computation of a minibatch. If indeed $\tau_1 \ll \tau_b$, and in particular

$$\tau_1 = \mathcal{O}\left(\frac{\sigma}{\beta}\tau_b\right), \tag{15}$$

MP+SVRG dominates both MP+AGD *and* RSAG: the τ_b term is then the dominant term in its parallel runtime, with an improvement over both MP+SVRG, yet the total computational cost (number of individual gradient computations) does not increase over the optimal cost of RSAG, and is thus much smaller than MP+AGD—we get a reduction in parallel runtime without any additional computational cost. In particular, with a minibatch of size b_{MP}, and under the assumption (15), we get a parallel runtime of

$$T_{\text{MP-SVRG}} = \tilde{\mathcal{O}}\left(\frac{\sigma(\phi(\mathbf{w}_0) - \phi^*)}{\varepsilon^2}\right), \tag{16}$$

with optimal computational cost $N(\varepsilon)$. The requirement (15) is very reasonable, especially for large b, considering the required ratio does not depend on b. All we require is that parallel computation of a gradient on a large minibatch is at least a constant factor more expensive than an individual gradient computation on one point.

4. Other parallel optimization frameworks, such as DANE (Shamir et al., 2014) and AIDE (Reddi et al., 2016), can be applied and analyzed similarly.

4. Discussion

In this work, we have focused on stochastic nonconvex optimization using only noisy first-order gradient information, and made a step toward large minibatch training. Our results suggest that it is beneficial to perform better optimization on each minibatch than a single gradient descent, when the minibatch size is too large to be sample-inefficient in minibatch SGD.

Unfortunately, we could not yet remove the "optimization error" altogether from the convergence rate as is achievable for stochastic convex optimization using minibatch-prox (Wang et al., 2017): in comparison to the convergence rate (3) by accelerated minibatch SGD, the minibatch-prox algorithm on a convex $\phi(\mathbf{w})$ provides the guarantee $\mathbb{E}\left[\phi(\mathbf{w}) - \phi(\mathbf{w}_*)\right] \leq \mathcal{O}\left(\frac{\|\mathbf{w}_0 - \mathbf{w}_*\|}{\sqrt{bK}}\right)$, so that one could use any minibatch size b while maintaining sample efficiency. Nor could we significantly reduce it as accelerated minibatch SGD achieved, again in the convex case (compare the first term in (3) and (4)). We also do not know if the "sample error" is the statistical limit for the class of problems considered here, and if a refined analysis of minibatch SGD (that makes use of the σ-almost convexity) can show the same convergence rate, which would tell if our more complicated algorithms are indeed necessary. In combination with the convexification procedure, Carmon et al. (2017) additionally made use of curvature information in the Hessian (which can be efficiently obtained for deep learning models, Pearlmutter, 1994; Martens, 2010) so as to further reduce the "optimization error": for their ERM algorithm, the number of gradient steps needed is $\mathcal{O}\left(\frac{1}{\varepsilon^{7/4}}\right)$, rather than $\mathcal{O}\left(\frac{1}{\varepsilon^2}\right)$ for minibatch SGD and our algorithms. We suspect that using the same technique in stochastic optimization may yield similar improvement, at the cost of a more complex algorithm.

Acknowledgement

Weiran Wang would like to thank Michael Maire for inspiring discussions on Goyal et al. (2017), and Zeyuan Allen-Zhu for helpful discussions on the convexification procedure.

Appendix A. Experiments

We now demonstrate our theory and the basic MP algorithm with an illustrative example. We train a neural network with 2 tanh hidden layers of 512 units each, and a softmax output layer to perform 10-way digit classification on the infinite MNIST dataset (Loosli et al., 2007). The dataset is randomly split into 8×10^6 samples for training and 10^5 samples for testing. To mimic the stochastic setting, we allow each method to load the training set into memory only once (this is equivalent to a single training epoch for minibatch SGD).

Performance of minibatch SGD We carefully tune the training hyperparameters by grid search for minibatch SGD with momentum, which remains a very strong method for training deep models in practice: the fixed learning rate is selected from $\{0.001, 0.01, 0.05, 0.1, 0.5\}$, and the momentum parameter from $\{0, 0.5, 0.8, 0.9, 0.99, 0.995\}$.

We vary the minibatch size b in $\{200, 1000, 2000, 10000\}$, and for each each b select the optimal combination of learning rate and momentum based on the objective on the test set. The test set objective vs. number of samples processed for different minibatch sizes are given in Figure 2 (left plot). We note that this type of learning curve (or error vs. epoch) is typically used for evaluating learning methods (e.g., Goyal et al., 2017), and is quite reasonable since the number of samples

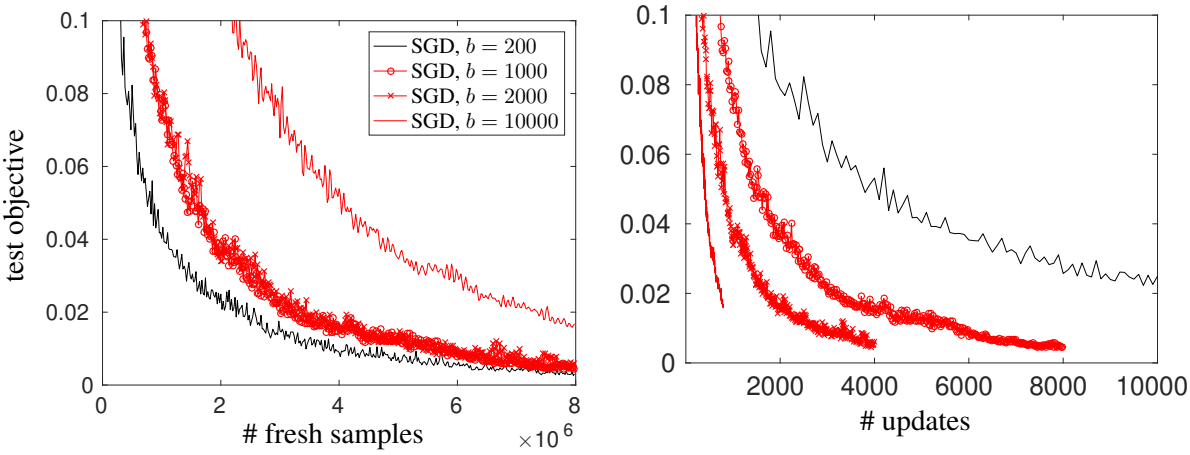

Figure 2: Performance of minibatch SGD with different minibatch sizes.

processed corresponds to the total energy spent. Observe that with smaller b, minibatch SGD converges to lower objective function values (although we have trained the neural network for only one epoch, the trained model with $b = 200$ has a cross-entropy loss of 0.0026 and a low classification error rate of 0.062% on the test set). The difference in final test objectives is small for $b = 200$ and $b = 1000$, but we start to see clear degradation of accuracy for $b = 10000$.

On the other hand, we provide test set objective vs. number of updates in Figure 2 (right plot). And we observe that the decrease of objective is much steeper for larger b, implying that a single gradient descent update with large b is of higher quality.

Performance of MP We now show that MP can achieve significantly higher accuracy with $b = 10000$, even slight improving over that of minibatch SGD with $b = 200$, using the same number of fresh samples and moderate number of gradient updates.

In our MP implementation, we approximately solve the subproblems on each minibatch with g gradient descent steps with momentum[5], yielding a training procedure similar to that of minibatch SGD, except that each large minibatch is kept in memory for g steps before switching to the next one, and that the gradient contains a *retraction* term $\gamma(\mathbf{w} - \mathbf{w}_{t-1})$ from the quadratic regularization. The learning rate and momentum parameter are tuned over a smaller grid around the optimal values for minibatch SGD. We tune g over $\{5,\ 10,\ 20,\ 50\}$ and the regularization parameter γ over the grid $\{0,\ 10^{-6},\ 10^{-4},\ 10^{-2},\ 1\}$. This implementation reduces to minibatch SGD when $\gamma = 0$ and $g = 1$.

For each γ, we select the combination of rest hyperparameters that gives the lowest test objective. Learning curves (objective vs. # fresh samples, and objective vs. # updates) for different values of γ are given in Figure 3, where we also compare with the learning curves of minibatch SGD at $b = 200$ and $b = 10000$. Observe that, with moderate values of g, MP can match the objective vs. # fresh samples curve of minibatch SGD at $b = 200$ so that it is sample-efficient (the trained model with $\gamma = 10^{-2}$ and $g = 50$ has a cross-entropy loss of 0.0004 and a classification error rate of 0.012%). On the other hand, MP is close to minibatch SGD at $b = 10000$ for the objective vs. # updates learning curve, so that each step is still of high quality and quickly decreases the objective.

5. Gradient descent with momentum and accelerated gradient descent have similar forms of updates.

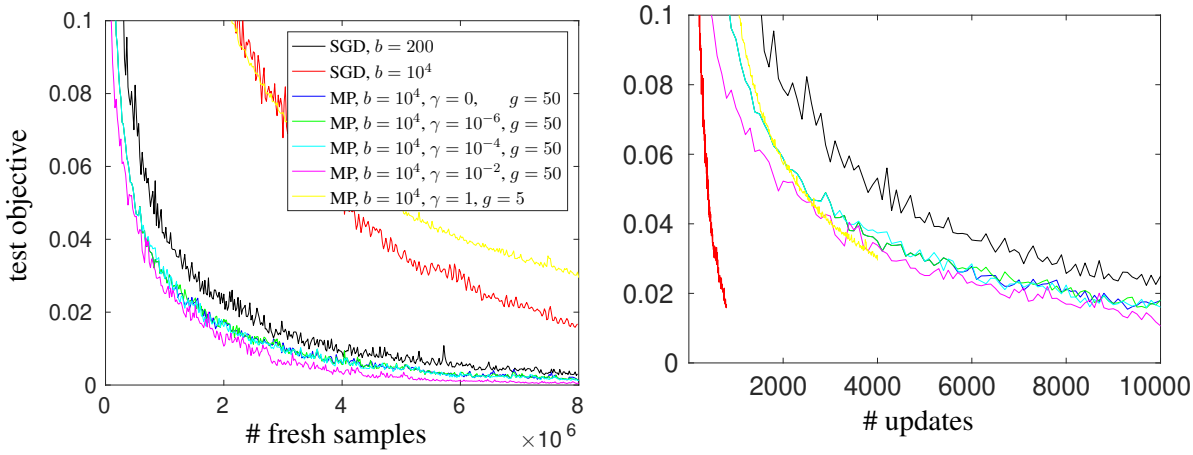

Figure 3: Performance of MP+AGD with different γ.

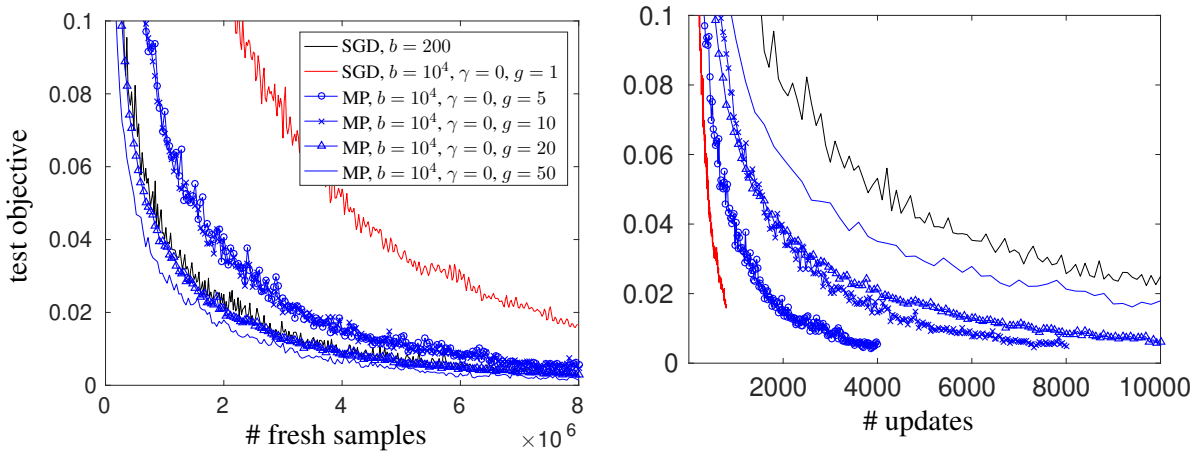

Figure 4: Performance of MP+AGD with $\gamma = 0$ and different number of gradient updates per minibatch g.

We have seen in Figure 3 that in fact $\gamma = 0$ works quite well without the retraction term (difference in final objectives are not significant for small γ), implying that simply processing the same large minibatch multiple times in minibatch SGD helps improve the sample efficiency. For this simple method, we provide the learning curves at different g values in Figure 4. From this figure, it is clear that we have speedup in terms of # gradient updates (or parallel runtime) at various levels of test objective (and hence classification error rate). For example, to obtain a test objective of 0.01 (roughly corresponding to an acceptable error rate of 0.32%), we can use MP with $b = 10000$ and $g = 5$ for about 2600 updates, while minibatch SGD with $b = 200$ obtains the same objective after about 17000 updates. This result demonstrates the success of a practical version of our method: if we have a constant stream of data, we can perform several gradient steps on each large minibatch in a parallel system to improve runtime, without losing much statistical precision.

Choice of hyperparameters We comment on how to select in practice the hyperparameters—minibatch size b, regularization parameter γ, and the number of gradient steps on each minibach g. Intuitively, one may set b to be as large as possible to fully utilize the parallel system, and set the number of steps g to be relatively small to use more fresh samples. Finally, γ is a type of stepsize and is better tuned on a validation set, as one would do for minibatch SGD. These are essentially the principles we followed in the above experiments.

Appendix B. A memory-efficient version of MP

We have shown in previous sections that it suffices to approximately minimize the stochastic objective $F_t(\mathbf{w})$ with $\mathcal{O}\left(\frac{1}{\varepsilon^2}\right)$ samples, by approximately minimizing the empirical objective $\hat{F}_t(\mathbf{w})$. But the number of samples can be too large (as $\varepsilon \to 0$) that the memory requirement is high, since we need to store this many samples and process them multiple times. In this section, we provide a modified algorithm to resolve this issue, which achieves the same learning guarantee with the same level of total samples, and using any (sufficiently large) minibatch size.

The modified algorithm is based on the analysis of minibatch-prox by Wang et al. (2017) for convex objectives. The authors showed that for Lipschitz and strongly convex stochastic objective, the minibatch-prox algorithm achieves the optimal $\mathcal{O}(1/n)$ rate[6] using n total samples and any minibatch size. We can therefore apply their results to the problem of $\min_{\mathbf{w}} F_t(\mathbf{w})$ at each iterations.

In this section, we use $F(\mathbf{x}) = \phi(\mathbf{x}) + \frac{\gamma}{2}\|\mathbf{x}-\mathbf{y}\|^2$ to denote the stochastic objective $F_t(\mathbf{w})$ at any iteration, which is $(\beta+\gamma)$-smooth and $(\gamma-\sigma)$-strongly convex in $\mathbf{x}$. The lemma below is parallel to Wang et al. (2017, Theorem 8). Its proof is also similar to theirs, with the difference being the stability used: theirs used stability for Lipschitz losses, whereas ours use the stability for smooth loss given in Lemma 3.

Lemma 7 *Assume the same conditions of Lemma 3 on the instantaneous loss. Consider the following iterative procedure: for $s = 1, \dots, S$*

$$\mathbf{x}_s \approx \hat{\mathbf{x}}_s = \arg\min_{\mathbf{x}} \ \hat{G}_s(\mathbf{x})$$

$$\text{where } \hat{G}_s(\mathbf{x}) := \hat{F}(\mathbf{x}) + \frac{\rho_s}{2}\|\mathbf{x}-\mathbf{x}_{s-1}\|^2 = \frac{1}{m}\sum_{i=1}^{m}\ell(\mathbf{x},\xi_i^s) + \frac{\gamma}{2}\|\mathbf{x}-\mathbf{y}\|^2 + \frac{\rho_s}{2}\|\mathbf{x}-\mathbf{x}_{s-1}\|^2,$$

where $\rho_s > 0$, and $Z_s = \{\xi_1^s, \dots, \xi_m^s\}$ are m i.i.d. samples drawn from the underlying distribution at iteration s. Let $\mathbf{x}_s$ be the output of a randomized algorithm $\mathcal{A}$ satisfying $\mathbb{E}_{Z_s,\mathcal{A}}\left[\hat{G}_s(\mathbf{x}_s) - \hat{G}_s(\hat{\mathbf{x}}_s)\right] \le \eta_s$. Then with the following choices of parameters:

$$m \ge \frac{2(\sigma+\beta)}{\gamma-\sigma}, \qquad \rho_s = \frac{(\gamma-\sigma)(s-1)}{2}, \qquad \eta_s = \frac{V^2 S}{(\beta+\gamma)m}\cdot\frac{1}{s^5},$$

we have for $\bar{\mathbf{x}}_S = \frac{2}{S(S+1)}\sum_{s=1}^{S} s\mathbf{x}_s$ that

$$\mathbb{E}\left[F(\bar{\mathbf{x}}_S) - F(\mathbf{x}^*)\right] \le \frac{200V^2}{(\gamma-\sigma)mS}.$$

6. This rate is optimal in the sense of Nemirovski and Yudin (1983) and Agarwal et al. (2012).

Algorithm 1 Memory-efficient minibatch-prox for stochastic nonconvex optimization $\min_{\mathbf{w}} \phi(\mathbf{w})$.

Initialize $\mathbf{w}_0$.
for $t = 1, 2, \ldots, K$ **do**
  // Approximately solve $\min_{\mathbf{w}} \phi(\mathbf{w}) + \frac{\gamma}{2} \|\mathbf{w} - \mathbf{w}_{t-1}\|^2$
  Intialize inner loop $\mathbf{x}_0^{(t)}$
  for $s = 1, 2, \ldots, S$ **do**
    Draw m fresh samples $Z_s^{(t)} = \left\{\xi_1^{t,s}, \ldots, \xi_m^{t,s}\right\}$
    Approximately compute

$$\mathbf{x}_s^{(t)} \leftarrow \min_{\mathbf{x}} \frac{1}{m} \sum_{i=1}^{m} \ell(\mathbf{x}, \xi_i^{t,s}) + \frac{\gamma}{2} \|\mathbf{x} - \mathbf{w}_{t-1}\|^2 + \frac{\rho_s}{2} \left\|\mathbf{x} - \mathbf{x}_{s-1}^{(t)}\right\|^2$$

    by accelerated gradient descent or finite-sum methods
  end for
  $\mathbf{w}_t \leftarrow \frac{2}{S(S+1)} \sum_{s=1}^{S} s\mathbf{x}_s^{(t)}$
end for
Output: Pick $R \in \{1, \ldots, K\}$ uniformly at random and return $\mathbf{w}_R$.

Let the total number of samples used in this procedure be $b = mS$. This lemma shows that the $\mathcal{O}\left(\frac{1}{b}\right)$ convergence rate for $F(\mathbf{x})$ (as in Lemma 3) is still achievable, by iteratively drawing smaller minibatches and solving one simpler ERM on each.

This approach leads to an algorithm with intuitively two levels of minibatch-prox, one for the convexification of nonconvex objective, and the other for memory efficiency. We provide the sketch of the resulting algorithm in Algorithm 1.

Appendix C. Proof of Lemma 1

Proof Due to the $(\beta + \gamma)$-smoothness of $F_t(\mathbf{w})$ and the optimality condition that $\nabla F_t(\mathbf{w}_t^*) = \mathbf{0}$, we have (Nesterov, 2004, Theorem 2.1.5)

$$\mathbb{E}_{\mathcal{A}} \|\nabla F_t(\mathbf{w}_t)\|^2 \leq 2(\beta + \gamma) \cdot \mathbb{E}_{\mathcal{A}} \left[F_t(\mathbf{w}_t) - F_t(\mathbf{w}_t^*)\right] \leq 2(\beta + \gamma)\epsilon.$$

Then, by the definition of $F_t(\mathbf{w})$, it holds that

$$\begin{aligned}
\mathbb{E}_{\mathcal{A}} \|\nabla \phi(\mathbf{w}_t)\|^2 &= \mathbb{E}_{\mathcal{A}} \|\nabla F_t(\mathbf{w}_t) - \gamma(\mathbf{w}_t - \mathbf{w}_{t-1})\|^2 \\
&\leq 2\mathbb{E}_{\mathcal{A}} \|\nabla F_t(\mathbf{w}_t)\|^2 + 2\gamma^2 \mathbb{E}_{\mathcal{A}} \|\mathbf{w}_{t-1} - \mathbf{w}_t\|^2 \\
&\leq 4(\beta + \gamma)\epsilon + 2\gamma^2 \mathbb{E}_{\mathcal{A}} \|\mathbf{w}_{t-1} - \mathbf{w}_t\|^2
\end{aligned}$$

where we have used the fact that $(x + y)^2 \leq 2(x^2 + y^2)$ in the first inequality. ■

Appendix D. Proof of Corollary 2

Proof By the definition of $F_t(\mathbf{w})$, we have

$$\begin{aligned}\mathbb{E}_{\mathcal{A}}\left[\phi(\mathbf{w}_{t-1}) - \phi(\mathbf{w}_t)\right] &= \mathbb{E}_{\mathcal{A}}\left[F_t(\mathbf{w}_{t-1}) - F_t(\mathbf{w}_t) + \frac{\gamma}{2}\left\|\mathbf{w}_{t-1} - \mathbf{w}_t\right\|^2\right] \\ &= F_t(\mathbf{w}_{t-1}) - F_t(\mathbf{w}_t^*) + \mathbb{E}_{\mathcal{A}}\left[F_t(\mathbf{w}_t^*) - F_t(\mathbf{w}_t) + \frac{\gamma}{2}\left\|\mathbf{w}_{t-1} - \mathbf{w}_t\right\|^2\right] \\ &\geq \frac{\gamma}{2}\mathbb{E}_{\mathcal{A}}\left\|\mathbf{w}_{t-1} - \mathbf{w}_t\right\|^2 - \epsilon\end{aligned}$$

where the expectation is taken over randomness at iteration t.

Averaging this inequality over $t = 1, \ldots, K$ and taking expectation over randomness in all iterations, we have

$$\frac{1}{K}\sum_{t=1}^{K}\mathbb{E}_{\mathcal{A}}\left\|\mathbf{w}_{t-1} - \mathbf{w}_t\right\|^2 \leq \frac{2\mathbb{E}_{\mathcal{A}}\left[\phi(\mathbf{w}_0) - \phi(\mathbf{w}_K)\right]}{\gamma K} + \frac{2\epsilon}{\gamma} \leq \frac{2(\phi(\mathbf{w}_0) - \phi^*)}{\gamma K} + \frac{2\epsilon}{\gamma}.$$

This implies that if we randomly pick $R \in \{1, \ldots, K\}$, it holds that

$$\mathbb{E}_{R,\mathcal{A}}\left\|\mathbf{w}_{R-1} - \mathbf{w}_R\right\|^2 \leq \frac{2(\phi(\mathbf{w}_0) - \phi^*)}{\gamma K} + \frac{2\epsilon}{\gamma}.$$

Plugging this into Lemma 1 yields the desired result. ■

Appendix E. Proof of Lemma 3

Proof The first part of this proof is adapted from that of Shalev-Shwartz and Ben-David (2014)[Section 13.3.2] for smooth and nonnegative losses. Note that our bound does not assume the nonnegativity of the instantaneous loss.

Exact ERM Denote by $Z^{(i)}$ the sample set that is identical to Z except that the i-th sample ξ_i is replaced by another random sample ξ_i', by $\hat{F}^{(i)}(\mathbf{w})$ the empirical objective defined using $Z^{(i)}$, i.e.,

$$\hat{F}^{(i)}(\mathbf{w}) := \frac{1}{b}\left(\sum_{j\neq i}\ell(\mathbf{w}, \xi_i) + \ell(\mathbf{w}, \xi_i')\right) + r(\mathbf{w}),$$

and by $\hat{\mathbf{w}}^{(i)} = \arg\min_{\mathbf{w}} \hat{F}^{(i)}(\mathbf{w})$ the empirical risk minimizer of $\hat{F}^{(i)}(\mathbf{w})$.

First, observe that

$$\begin{aligned}\hat{F}(\hat{\mathbf{w}}^{(i)}) - \hat{F}(\hat{\mathbf{w}}) &= \frac{\ell(\hat{\mathbf{w}}^{(i)}, \xi_i) - \ell(\hat{\mathbf{w}}, \xi_i)}{b} + \frac{\sum_{j\neq i}\ell(\hat{\mathbf{w}}^{(i)}, \xi_j) - \ell(\hat{\mathbf{w}}, \xi_j)}{b} + r(\hat{\mathbf{w}}^{(i)}) - r(\hat{\mathbf{w}}) \\ &= \frac{\ell(\hat{\mathbf{w}}^{(i)}, \xi_i) - \ell(\hat{\mathbf{w}}, \xi_i)}{b} + \frac{\ell(\hat{\mathbf{w}}, \xi_i') - \ell(\hat{\mathbf{w}}^{(i)}, \xi_i')}{b} + \left(\hat{F}^{(i)}(\hat{\mathbf{w}}^{(i)}) - \hat{F}^{(i)}(\hat{\mathbf{w}})\right) \\ &\leq \frac{\ell(\hat{\mathbf{w}}^{(i)}, \xi_i) - \ell(\hat{\mathbf{w}}, \xi_i)}{b} + \frac{\ell(\hat{\mathbf{w}}, \xi_i') - \ell(\hat{\mathbf{w}}^{(i)}, \xi_i')}{b} - \frac{\gamma - \sigma}{2}\left\|\hat{\mathbf{w}}^{(i)} - \hat{\mathbf{w}}\right\|^2\end{aligned}$$

where we have used in the inequality the fact that $\hat{\mathbf{w}}^{(i)}$ is the minimizer of $\hat{F}^{(i)}(\mathbf{w})$, which is $(\gamma - \sigma)$-strongly convex.

On the other hand, it follows from the $(\gamma-\sigma)$-strong convexity of $\hat{F}(\mathbf{w})$ that

$$\hat{F}(\hat{\mathbf{w}}^{(i)}) - \hat{F}(\hat{\mathbf{w}}) \geq \frac{\gamma-\sigma}{2}\left\|\hat{\mathbf{w}}^{(i)} - \hat{\mathbf{w}}\right\|^2.$$

Combining the above two inequalities, and applying the σ-almost convexity and β-smoothness of $\ell(\mathbf{w},\xi)$, we obtain

$$\begin{aligned}
&(\gamma-\sigma)b\cdot\left\|\hat{\mathbf{w}}^{(i)} - \hat{\mathbf{w}}\right\|^2 \\
\leq&\left(\ell(\hat{\mathbf{w}}^{(i)},\xi_i) - \ell(\hat{\mathbf{w}},\xi_i)\right) + \left(\ell(\hat{\mathbf{w}},\xi_i') - \ell(\hat{\mathbf{w}}^{(i)},\xi_i')\right) \qquad (17)\\
\leq&\langle\nabla\ell(\hat{\mathbf{w}}^{(i)},\xi_i),\,\hat{\mathbf{w}}^{(i)} - \hat{\mathbf{w}}\rangle + \frac{\sigma}{2}\left\|\hat{\mathbf{w}}^{(i)} - \hat{\mathbf{w}}\right\|^2 + \langle\nabla\ell(\hat{\mathbf{w}},\xi_i'),\,\hat{\mathbf{w}} - \hat{\mathbf{w}}^{(i)}\rangle + \frac{\sigma}{2}\left\|\hat{\mathbf{w}} - \hat{\mathbf{w}}^{(i)}\right\|^2\\
=&\langle\nabla\ell(\hat{\mathbf{w}}^{(i)},\xi_i) - \nabla\ell(\hat{\mathbf{w}},\xi_i'),\,\hat{\mathbf{w}}^{(i)} - \hat{\mathbf{w}}\rangle + \sigma\left\|\hat{\mathbf{w}}^{(i)} - \hat{\mathbf{w}}\right\|^2\\
=&\langle\nabla\ell(\hat{\mathbf{w}}^{(i)},\xi_i) - \nabla\phi(\hat{\mathbf{w}}^{(i)}),\,\hat{\mathbf{w}}^{(i)} - \hat{\mathbf{w}}\rangle + \langle\nabla\phi(\hat{\mathbf{w}}^{(i)}) - \nabla\phi(\hat{\mathbf{w}}),\,\hat{\mathbf{w}}^{(i)} - \hat{\mathbf{w}}\rangle\\
&\quad+\langle\nabla\phi(\hat{\mathbf{w}}) - \nabla\ell(\hat{\mathbf{w}},\xi_i'),\,\hat{\mathbf{w}}^{(i)} - \hat{\mathbf{w}}\rangle + \sigma\left\|\hat{\mathbf{w}}^{(i)} - \hat{\mathbf{w}}\right\|^2\\
\leq&\left(\left\|\nabla\ell(\hat{\mathbf{w}}^{(i)},\xi_i) - \nabla\phi(\hat{\mathbf{w}}^{(i)})\right\| + \left\|\nabla\ell(\hat{\mathbf{w}},\xi_i') - \nabla\phi(\hat{\mathbf{w}})\right\|\right)\cdot\left\|\hat{\mathbf{w}}^{(i)} - \hat{\mathbf{w}}\right\| + (\sigma+\beta)\left\|\hat{\mathbf{w}}^{(i)} - \hat{\mathbf{w}}\right\|^2. \qquad (18)
\end{aligned}$$

By the assumption that $\sigma+\beta \leq \frac{(\gamma-\sigma)b}{2}$, we then have

$$\begin{aligned}
\left\|\hat{\mathbf{w}}^{(i)} - \hat{\mathbf{w}}\right\| &\leq \frac{1}{(\gamma-\sigma)b - (\sigma+\beta)}\left(\left\|\nabla\ell(\hat{\mathbf{w}}^{(i)},\xi_i) - \nabla\phi(\hat{\mathbf{w}}^{(i)})\right\| + \left\|\nabla\ell(\hat{\mathbf{w}},\xi_i') - \nabla\phi(\hat{\mathbf{w}})\right\|\right)\\
&\leq \frac{2}{(\gamma-\sigma)b}\left(\left\|\nabla\ell(\hat{\mathbf{w}}^{(i)},\xi_i) - \nabla\phi(\hat{\mathbf{w}}^{(i)})\right\| + \left\|\nabla\ell(\hat{\mathbf{w}},\xi_i') - \nabla\phi(\hat{\mathbf{w}})\right\|\right).
\end{aligned}$$

Taking expectations of (17) and (18) over the samples and plugging in the above inequality yields

$$\begin{aligned}
&2\mathbb{E}_{Z\cup\{\xi_i'\}}\left[\phi(\hat{\mathbf{w}}) - \hat{\phi}(\hat{\mathbf{w}})\right]\\
\leq&\left(\frac{2}{(\gamma-\sigma)b} + \frac{4(\sigma+\beta)}{(\gamma-\sigma)^2b^2}\right)\cdot\mathbb{E}_{Z\cup\{\xi_i'\}}\left(\left\|\nabla\ell(\hat{\mathbf{w}}^{(i)},\xi_i) - \nabla\phi(\hat{\mathbf{w}}^{(i)})\right\| + \left\|\nabla\ell(\hat{\mathbf{w}},\xi_i') - \nabla\phi(\hat{\mathbf{w}})\right\|\right)^2\\
\leq&\frac{4}{(\gamma-\sigma)b}\mathbb{E}_{Z\cup\{\xi_i'\}}\left(\left\|\nabla\ell(\hat{\mathbf{w}}^{(i)},\xi_i) - \nabla\phi(\hat{\mathbf{w}}^{(i)})\right\| + \left\|\nabla\ell(\hat{\mathbf{w}},\xi_i') - \nabla\phi(\hat{\mathbf{w}})\right\|\right)^2\\
\leq&\frac{8}{(\gamma-\sigma)b}\left(\mathbb{E}_{Z^{(i)}}\left[\mathbb{E}_{\{\xi_i\}}\left\|\nabla\ell(\hat{\mathbf{w}}^{(i)},\xi_i) - \nabla\phi(\hat{\mathbf{w}}^{(i)})\right\|^2\right] + \mathbb{E}_{Z}\left[\mathbb{E}_{\{\xi_i'\}}\left\|\nabla\ell(\hat{\mathbf{w}},\xi_i') - \nabla\phi(\hat{\mathbf{w}})\right\|^2\right]\right)\\
\leq&\frac{16V^2}{(\gamma-\sigma)b}
\end{aligned}$$

where we have used the triangle inequality in the first step, the assumption $2(\sigma+\beta)\leq(\gamma-\sigma)b$ in the second step, the fact that $(x+y)^2\leq 2(x^2+y^2)$ in the third step, and the assumption on the variance of stochastic gradient in the final step.

Since $\hat{\mathbf{w}}$ minimizes $\hat{F}(\mathbf{w})$, we have $\hat{F}(\hat{\mathbf{w}}) \leq \hat{F}(\mathbf{w}^*)$ and consequently

$$\mathbb{E}_Z\left[F(\hat{\mathbf{w}})\right] = \mathbb{E}_Z\left[\hat{F}(\hat{\mathbf{w}}) + \phi(\hat{\mathbf{w}}) - \hat{\phi}(\hat{\mathbf{w}})\right] \leq F(\mathbf{w}^*) + \mathbb{E}_Z\left[\phi(\hat{\mathbf{w}}) - \hat{\phi}(\hat{\mathbf{w}})\right] \leq F(\mathbf{w}^*) + \frac{8V^2}{(\gamma-\sigma)b}$$

Then the first part of the lemma follows.

Inexact ERM For the approximate solution $\tilde{\mathbf{w}}$, due to the $(\gamma-\sigma)$-strong convexity of $\hat{F}(\mathbf{w})$, we have

$$\mathbb{E}_{Z,\mathcal{A}}\left\|\tilde{\mathbf{w}} - \hat{\mathbf{w}}\right\|^2 \leq \frac{2}{\gamma-\sigma}\mathbb{E}_{Z,\mathcal{A}}\left[\hat{F}(\tilde{\mathbf{w}}) - \hat{F}(\hat{\mathbf{w}})\right] \leq \frac{2\delta}{\gamma-\sigma}.$$

Now, in view of the $(\beta+\gamma)$-smoothness of $F(\mathbf{w})$, it holds that

$$\begin{aligned}
\mathbb{E}_{Z,\mathcal{A}}\left[F(\tilde{\mathbf{w}}) - F(\hat{\mathbf{w}})\right] &\leq \mathbb{E}_{Z,\mathcal{A}}\langle \nabla F(\hat{\mathbf{w}}),\, \tilde{\mathbf{w}} - \hat{\mathbf{w}}\rangle + \frac{\beta+\gamma}{2}\mathbb{E}_{Z,\mathcal{A}}\left\|\tilde{\mathbf{w}} - \hat{\mathbf{w}}\right\|^2 \\
&\leq \mathbb{E}_{Z,\mathcal{A}}\left[\|\nabla F(\hat{\mathbf{w}})\| \cdot \|\tilde{\mathbf{w}} - \hat{\mathbf{w}}\|\right] + \frac{\beta+\gamma}{2}\mathbb{E}_{Z,\mathcal{A}}\left\|\tilde{\mathbf{w}} - \hat{\mathbf{w}}\right\|^2 \\
&\leq \mathbb{E}_{Z,\mathcal{A}}\left[\sqrt{2(\beta+\gamma)\cdot(F(\hat{\mathbf{w}}) - F(\mathbf{w}^*))}\left\|\tilde{\mathbf{w}} - \hat{\mathbf{w}}\right\|\right] + \frac{\beta+\gamma}{2}\mathbb{E}_{Z,\mathcal{A}}\left\|\tilde{\mathbf{w}} - \hat{\mathbf{w}}\right\|^2 \\
&\leq \mathbb{E}_Z\left[F(\hat{\mathbf{w}}) - F(\mathbf{w}^*)\right] + (\beta+\gamma)\cdot\mathbb{E}_{Z,\mathcal{A}}\left\|\tilde{\mathbf{w}} - \hat{\mathbf{w}}\right\|^2 \\
&\leq \frac{8V^2}{(\gamma-\sigma)b} + \frac{2(\beta+\gamma)\delta}{\gamma-\sigma}
\end{aligned}$$

where we have used Nesterov (2004, Theorem 2.1.5) in the third inequality, and the fact that $xy \leq x^2 + \frac{y^2}{4}$ in the fourth inequality. Then the lemma follows by combining this inequality with the stability of exact ERM. ■

Appendix F. Proof of Theorem 5

Proof Let $\delta = \frac{8V^2}{(\beta+\gamma)b}$, then by the second part of Lemma 3, it holds that

$$\mathbb{E}_{Z_t,\mathcal{A}}\left[F_t(\mathbf{w}_t) - F_t(\mathbf{w}_t^*)\right] \leq \frac{32V^2}{(\gamma-\sigma)b}.$$

Combining this with Corollary 2 yields

$$\begin{aligned}
\mathbb{E}_{R,\mathcal{A}}\left\|\nabla\phi(\mathbf{w}_R)\right\|^2 &\leq \frac{4\gamma\left(\phi(\mathbf{w}_0) - \phi^*\right)}{K} + (4\beta + 8\gamma)\cdot\frac{32V^2}{(\gamma-\sigma)b} \\
= \frac{4\sigma\left(\phi(\mathbf{w}_0) - \phi^*\right)}{K} + \frac{256V^2}{b} + \frac{4(\gamma-\sigma)\left(\phi(\mathbf{w}_0) - \phi^*\right)}{K} &+ (4\beta + 8\sigma)\cdot\frac{32V^2}{(\gamma-\sigma)b}.
\end{aligned}$$

Minimizing the right hand side over γ yields the optimal choice $\gamma = \sigma + \sqrt{\frac{32(\beta+2\sigma)V^2K}{(\phi(\mathbf{w}_0)-\phi^*)b}}$ and the desired result. ■

Appendix G. Proof of Lemma 7

Proof Denote by $\mathbf{x}^* = \arg\min_{\mathbf{x}} F(\mathbf{x})$ the unique minimizer of $F(\mathbf{x})$, and by $G_s(\mathbf{x}) = \phi(\mathbf{x}) + \frac{\gamma}{2}\|\mathbf{x}-\mathbf{y}\|^2 + \frac{\rho_s}{2}\|\mathbf{x}-\mathbf{x}_{s-1}\|^2$ the population counterpart of $\hat{G}_s(\mathbf{x})$, with unique minimizer $\mathbf{x}_s^* = \arg\min_{\mathbf{x}} G_s(\mathbf{x})$. In the following, we also use the shorthand $L = \beta + \gamma$, and $\lambda = \gamma - \sigma$.

First, by the $(\lambda+\rho_s)$-strong convexity of $\hat{G}_s(\mathbf{x})$, we have

$$\hat{G}_s(\mathbf{x}^*) \geq \hat{G}_s(\hat{\mathbf{x}}_s) + \frac{\lambda+\rho_s}{2}\|\mathbf{x}^* - \hat{\mathbf{x}}_s\|^2. \tag{19}$$

By the first part of Lemma 3 (we are now applying the lemma to $\hat{G}_s(\mathbf{x})$ and $G_s(\mathbf{x})$, whose data-independent regularizer $\frac{\gamma}{2}\|\mathbf{x}-\mathbf{y}\|^2 + \frac{\rho_s}{2}\|\mathbf{x}-\mathbf{x}_{s-1}\|^2$ is $(\gamma+\rho_s)$-strongly convex), we have that $\mathbb{E}_{Z_s}\left[\phi(\mathbf{x}_s^*) - \frac{1}{m}\sum_{i=1}^m \ell(\mathbf{x}_s^*, \xi_i^s)\right] \leq \frac{8V^2}{(\lambda+\rho_s)m}$ as long as $m \geq \frac{2(\sigma+\beta)}{\lambda+\rho_s}$. Therefore, taking expectation of (19) over Z_s yields

$$\begin{aligned} F(\mathbf{x}^*) + \frac{\rho_s}{2}\|\mathbf{x}^* - \mathbf{x}_{s-1}\|^2 &\geq \mathbb{E}_{Z_s}\left[F(\hat{\mathbf{x}}_s) + \frac{\rho_s}{2}\|\hat{\mathbf{x}}_s - \mathbf{x}_{s-1}\|^2 - \frac{8V^2}{(\lambda+\rho_s)m} + \frac{\lambda+\rho_s}{2}\|\mathbf{x}^* - \hat{\mathbf{x}}_s\|^2\right] \\ &\geq \mathbb{E}_{Z_s}\left[F(\mathbf{x}_s) + \frac{\rho_s}{2}\|\mathbf{x}_s - \mathbf{x}_{s-1}\|^2 - \frac{16V^2}{(\lambda+\rho_s)m} - \frac{2(L+\rho_s)\eta_s}{\lambda+\rho_s} + \frac{\lambda+\rho_s}{2}\|\mathbf{x}^* - \hat{\mathbf{x}}_s\|^2\right] \end{aligned} \tag{20}$$

where we have used the second part of Lemma 3 in the second inequality.

Next, we relate $\hat{\mathbf{x}}_s$ to $\mathbf{x}_s$ for the last term of (20). By the $(\lambda+\rho_s)$-strong convexity of $\hat{G}_s(\mathbf{x})$, we have $\mathbb{E}_{Z_s,\mathcal{A}}\|\hat{\mathbf{x}}_s - \mathbf{x}_s\|^2 \leq \frac{2\eta_s}{\lambda+\rho_s}$, and then by the triangle inequality

$$\begin{aligned} \mathbb{E}_{Z_s,\mathcal{A}}\|\mathbf{x}^* - \hat{\mathbf{x}}_s\|^2 &\geq \mathbb{E}_{Z_s,\mathcal{A}}\left|\|\mathbf{x}^* - \mathbf{x}_s\| - \|\hat{\mathbf{x}}_s - \mathbf{x}_s\|\right|^2 \\ &\geq \mathbb{E}_{Z_s,\mathcal{A}}\|\mathbf{x}^* - \mathbf{x}_s\|^2 - 2\mathbb{E}_{Z_s,\mathcal{A}}\left[\|\mathbf{x}^* - \mathbf{x}_s\| \cdot \|\hat{\mathbf{x}}_s - \mathbf{x}_s\|\right] \\ &\geq \mathbb{E}_{Z_s,\mathcal{A}}\|\mathbf{x}^* - \mathbf{x}_s\|^2 - 2\sqrt{\mathbb{E}_{Z_s,\mathcal{A}}\|\mathbf{x}^* - \mathbf{x}_s\|^2}\sqrt{\mathbb{E}_{Z_s,\mathcal{A}}\|\hat{\mathbf{x}}_s - \mathbf{x}_s\|^2} \\ &\geq \mathbb{E}_{Z_s,\mathcal{A}}\|\mathbf{x}^* - \mathbf{x}_s\|^2 - 2\sqrt{\mathbb{E}_{Z_s,\mathcal{A}}\|\mathbf{x}^* - \mathbf{x}_s\|^2}\sqrt{\frac{2\eta_s}{\lambda+\rho_s}} \end{aligned} \tag{21}$$

where the third inequality is due to the Cauchy-Schwarz inequality.

Substituting (21) into (20) and rearranging terms, we obtain

$$\begin{aligned} \mathbb{E}_{Z_s,\mathcal{A}}\left[F(\mathbf{x}_s) - F(\mathbf{x}^*)\right] \leq \mathbb{E}_{Z_s,\mathcal{A}}&\left[\frac{\rho_s}{2}\|\mathbf{x}^* - \mathbf{x}_{s-1}\|^2 - \frac{\lambda+\rho_s}{2}\|\mathbf{x}^* - \mathbf{x}_s\|^2\right] \\ &+ \frac{16V^2}{(\lambda+\rho_s)m} + \frac{2(L+\rho_s)\eta_s}{\lambda+\rho_s} + \sqrt{2(\lambda+\rho_s)\eta_s}\sqrt{\mathbb{E}_{Z_s,\mathcal{A}}\|\mathbf{x}^* - \mathbf{x}_s\|^2}. \end{aligned}$$

Setting $\rho_s = \frac{\lambda(s-1)}{2}$, and multiplying both sides by s, we further obtain

$$\begin{aligned} s\mathbb{E}_{Z_s,\mathcal{A}}\left[F(\mathbf{x}_s) - F(\mathbf{x}^*)\right] \leq \mathbb{E}_{Z_s,\mathcal{A}}&\left[\frac{\lambda(s-1)s}{4}\|\mathbf{x}^* - \mathbf{x}_{s-1}\|^2 - \frac{\lambda s(s+1)}{4}\|\mathbf{x}^* - \mathbf{x}_s\|^2\right] \\ &+ \frac{32V^2}{\lambda m} + \frac{(4L+\lambda s)\eta_s}{\lambda} + \sqrt{\lambda s\eta_s}\sqrt{\mathbb{E}_{Z_s,\mathcal{A}}\left[s(s+1)\|\mathbf{x}^* - \mathbf{x}_s\|^2\right]}. \end{aligned}$$

Summing the above inequality over $s = 1, \ldots, S$ yields

$$\mathbb{E}\left[\sum_{s=1}^{S} s\left(F(\mathbf{x}_s) - F(\mathbf{x}^*)\right)\right] + \mathbb{E}\left[\frac{S(S+1)\left\|\mathbf{x}^* - \mathbf{x}_S\right\|^2}{4}\right]$$
$$\leq \frac{32V^2S}{\lambda m} + \frac{4L}{\lambda}\sum_{s=1}^{S}\eta_s + \sum_{s=1}^{S} s\eta_s + \sum_{s=1}^{S}\sqrt{\lambda s\eta_s}\sqrt{\mathbb{E}\left[s(s+1)\left\|\mathbf{x}^* - \mathbf{x}_s\right\|^2\right]}. \tag{22}$$

Bounding $\|\mathbf{x}^* - \mathbf{x}_S\|^2$ Dropping the $\mathbb{E}\left[\sum_{s=1}^{S} s\left(F(\mathbf{x}_s) - F(\mathbf{x}^*)\right)\right]$ term from (22) which is nonnegative, we have

$$\mathbb{E}\left[S(S+1)\left\|\mathbf{x}^* - \mathbf{x}_S\right\|^2\right]$$
$$\leq \frac{128V^2S}{\lambda m} + \frac{16L}{\lambda}\sum_{s=1}^{S}\eta_s + 4\sum_{s=1}^{S} s\eta_s + 4\sum_{s=1}^{S}\sqrt{\lambda s\eta_s}\sqrt{\mathbb{E}\left[s(s+1)\left\|\mathbf{x}^* - \mathbf{x}_s\right\|^2\right]}.$$

Now apply Lemma 8 and we obtain

$$\mathbb{E}\left[S(S+1)\left\|\mathbf{x}^* - \mathbf{x}_S\right\|^2\right] \leq \sqrt{\frac{128V^2S}{\lambda m}} + \sqrt{\frac{16L}{\lambda}\sum_{s=1}^{S}\eta_s} + \sqrt{4\sum_{s=1}^{S} s\eta_s} + 4\sum_{s=1}^{S}\sqrt{\lambda s\eta_s}.$$

Note that this bound increases with S.

Bounding the function value Dropping the $\mathbb{E}\left[S(S+1)\left\|\mathbf{x}^* - \mathbf{x}_S\right\|^2\right]$ term from (22) which is nonnegative, we have

$$\mathbb{E}\left[\sum_{s=1}^{S} s\left(F(\mathbf{x}_s) - F(\mathbf{x}^*)\right)\right]$$
$$\leq \frac{32V^2S}{\lambda m} + \frac{4L}{\lambda}\sum_{s=1}^{S}\eta_s + \sum_{s=1}^{S} s\eta_s + \sum_{s=1}^{S}\sqrt{\lambda s\eta_s}\sqrt{\mathbb{E}\left[S(S+1)\left\|\mathbf{x}^* - \mathbf{x}_S\right\|^2\right]}$$
$$\leq \frac{32V^2S}{\lambda m} + \frac{4L}{\lambda}\sum_{s=1}^{S}\eta_s + \sum_{s=1}^{S} s\eta_s$$
$$+ \left(\sum_{s=1}^{S}\sqrt{\lambda s\eta_s}\right)\left(\sqrt{\frac{128V^2S}{\lambda m}} + \sqrt{\frac{16L}{\lambda}\sum_{s=1}^{S}\eta_s} + \sqrt{4\sum_{s=1}^{S} s\eta_s} + 4\sum_{s=1}^{S}\sqrt{\lambda s\eta_s}\right). \tag{23}$$

We require that η_s decays with s, and in particular

$$\eta_s = \frac{V^2S}{Lm} \cdot \frac{1}{s^5}. \tag{24}$$

Recall that $\sum_{s=1}^{\infty} \frac{1}{s^{1+\delta}} \leq \frac{1+\delta}{\delta}$. Then (24) ensures

$$\sum_{s=1}^{S}\eta_s \leq 2\frac{V^2S}{Lm}, \qquad \sum_{s=1}^{S}\sqrt{\lambda s\eta_s} \leq 2\sqrt{\frac{\lambda V^2S}{Lm}}, \qquad \text{and} \qquad \sqrt{\sum_{s=1}^{S} s\eta_s} \leq 2\sqrt{\frac{V^2S}{Lm}}.$$

Plugging them into (23), and noting that $\lambda \leq L$, we obtain

$$\mathbb{E}\left[\sum_{s=1}^{S} s\left(F(\mathbf{x}_s) - F(\mathbf{x}^*)\right)\right] \leq \frac{100V^2 S}{\lambda m}.$$

By returning the weighted average $\bar{\mathbf{x}}_S = \frac{2}{S(S+1)}\sum_{s=1}^{S} s\mathbf{x}_s$ and the convexity of $F(\mathbf{x})$, we obtain the desired result. ■

Appendix H. An auxiliary lemma

Lemma 8 *(Schmidt et al., 2011, Lemma 1) Assume that the non-negative sequence $\{u_S\}$ satisfies the following recursion for all $S \geq 1$:*

$$u_S^2 \leq A_S + \sum_{s=1}^{S} \lambda_s u_s,$$

with A_S an increasing sequence, $A_0 \geq u_0^2$ and $\lambda_s \geq 0$ for all s. Then, for all $S \geq 1$, we have

$$u_S \leq \frac{1}{2}\sum_{s=1}^{S}\lambda_s + \left(A_S + \left(\frac{1}{2}\sum_{s=1}^{S}\lambda_s\right)^2\right)^{\frac{1}{2}} \leq \sqrt{A_S} + \sum_{s=1}^{S}\lambda_s.$$

References

Alekh Agarwal, Peter L. Bartlett, Pradeep Ravikumar, and Martin J. Wainwright. Information-theoretic lower bounds on the oracle complexity of stochastic convex optimization. *IEEE Trans. Information Theory*, 58(5):3235–3249, 2012.

Zeyuan Allen-Zhu. Natasha: Faster non-convex stochastic optimization via strongly non-convex parameter. In Doina Precup and Yee Whye Teh, editors, *Proc. of the 34rd Int. Conf. Machine Learning (ICML 2017)*, Sydney, Australia, August 6–11 2017.

D. P. Bertsekas. Convexification procedures and decomposition methods for nonconvex optimization problems. *J. Optimization Theory and Applications*, 29(2):169–197, 1979. URL http://web.mit.edu/dimitrib/www/Convexification_Mult.pdf.

Dimitri P. Bertsekas. *Nonlinear Programming*. Athena Scientific, Nashua, NH, second edition, 1999.

Dimitri P. Bertsekas. Incremental aggregated proximal and augmented Lagrangian algorithms. arXiv:1509.09257 [cs.SY], November 4 2015.

L. Bottou. Stochastic gradient learning in neural networks. In *Proc. Neuronîmes*, 1991.

Olivier Bousquet and André Elisseeff. Stability and generalization. *Journal of Machine Learning Research*, 2:499–526, March 2002.

Yair Carmon, John C. Duchi, Oliver Hinder, and Aaron Sidford. Accelerated methods for nonconvex optimization. arXiv:1611.00756 [math.OC], February 2 2017.

Andrew Cotter, Ohad Shamir, Nati Srebro, and Karthik Sridharan. Better mini-batch algorithms via accelerated gradient methods. In J. Shawe-Taylor, R. S. Zemel, P. Bartlett, F. Pereira, and K. Q. Weinberger, editors, *Advances in Neural Information Processing Systems (NIPS)*, volume 24, pages 1647–1655. MIT Press, Cambridge, MA, 2011.

Koby Crammer, Ofer Dekel, Joseph Keshet, Shai Shalev-Shwartz, and Yoram Singer. Online passive-aggressive algorithms. *Journal of Machine Learning Research*, 7:551–585, 2006.

Aaron Defazio. A simple practical accelerated method for finite sums. In Daniel D. Lee, Ulrike von Luxburg, and Isabelle Guyon, editors, *Advances in Neural Information Processing Systems (NIPS)*, volume 29. MIT Press, Cambridge, MA, 2016.

Ofer Dekel, Ran Gilad-Bachrach, Ohad Shamir, and Lin Xiao. Optimal distributed online prediction using mini-batches. *Journal of Machine Learning Research*, 13:165–202, 2012.

John Duchi, Elad Hazan, and Yoram Singer. Adaptive subgradient methods for online learning and stochastic optimization. *Journal of Machine Learning Research*, 12:2121–2159, July 2011.

Saeed Ghadimi and Guanghui Lan. Accelerated gradient methods for nonconvex nonlinear and stochastic programming. *Math. Prog.*, 156(1):59–99, 2016.

Saeed Ghadimi, Guanghui Lan, and Hongchao Zhang. Mini-batch stochastic approximation methods for nonconvex stochastic composite optimization. *Math. Prog.*, 155(1–2):267–305, 2016.

Priya Goyal, Piotr Dollár, Ross Girshick, Pieter Noordhuis, Lukasz Wesolowski, Aapo Kyrola, Andrew Tulloch, Yangqing Jia, and Kaiming He. Accurate, large minibatch SGD: Training imagenet in 1 hour. arXiv:1706.02677 [cs.CV], June 8 2017.

Elad Hoffer, Itay Hubara, and Daniel Soudry. Train longer, generalize better: Closing the generalization gap in large batch training of neural networks. arXiv:1705.08741 [stat.ML], May 24 2017.

Diederik Kingma and Jimmy Ba. Adam: A method for stochastic optimization. In *Proc. of the 3rd Int. Conf. Learning Representations (ICLR 2015)*, San Diego, CA, May 7–9 2015.

Brian Kulis and Peter L. Bartlett. Implicit online learning. In Johannes Fürnkranz and Thorsten Joachims, editors, *Proc. of the 27th Int. Conf. Machine Learning (ICML 2010)*, pages 575–582, Haifa, Israel, June 21–25 2010.

Guanghui Lan. An optimal method for stochastic composite optimization. *Math. Prog.*, 133(1–2): 365–397, June 2012.

Yann LeCun, Leon Bottou, Genevieve B. Orr, and Klaus-Robert Müller. Efficient backprop. volume 1524 of *Lecture Notes in Computer Science*, pages 9–50, Berlin, 1998. Springer-Verlag.

Jason D. Lee, Qihang Lin, Tengyu Ma, and Tianbao Yang. Distributed stochastic variance reduced gradient methods and a lower bound for communication complexity. arXiv:1507.07595 [math.OC], January 6 2016.

Mu Li, Tong Zhang, Yuqiang Chen, and Alexander J. Smola. Efficient mini-batch training for stochastic optimization. In *Proc. of the 20th ACM SIGKDD Int. Conf. Knowledge Discovery and Data Mining (SIGKDD 2014)*, pages 661–670, New York City, NY, August 24–27 2014.

Hongzhou Lin, Julien Mairal, and Zaid Harchaoui. A universal catalyst for first-order optimization. In C. Cortes, N. D. Lawrence, D. D. Lee, M. Sugiyama, and R. Garnett, editors, *Advances in Neural Information Processing Systems (NIPS)*, volume 28, pages 3366–3374. MIT Press, Cambridge, MA, 2015.

Gaëlle Loosli, Stéphane Canu, and Léon Bottou. Training invariant support vector machines using selective sampling. In Léon Bottou, Olivier Chapelle, Dennis DeCoste, and Jason Weston, editors, *Large Scale Kernel Machines*, Neural Information Processing Series, pages 301–320. MIT Press, 2007.

James Martens. Deep learning via Hessian-free optimization. In Johannes Fürnkranz and Thorsten Joachims, editors, *Proc. of the 27th Int. Conf. Machine Learning (ICML 2010)*, pages 735–742, Haifa, Israel, June 21–25 2010.

A. S. Nemirovski and D. B. Yudin. *Problem Complexity and Method Efficiency in Optimization*. John Wiley & Sons, 1983.

Y. Nesterov. *Introductory Lectures on Convex Optimization. A Basic Course*. Number 87 in Applied Optimization. Springer-Verlag, 2004.

Barak A. Pearlmutter. Fast exact multiplication by the Hessian. *Neural Computation*, 6(1):147–160, January 1994.

Sashank J. Reddi, Jakub Konecny, Peter Richtarik, Barnabas Poczos, and Alex Smola. AIDE: Fast and communication efficient distributed optimization. arXiv:1608.06879 [math.OC], August 24 2016.

Mark Schmidt, Nicolas Le Roux, and Francis Bach. Convergence rates of inexact proximal-gradient methods for convex optimization. In J. Shawe-Taylor, R. S. Zemel, P. Bartlett, F. Pereira, and K. Q. Weinberger, editors, *Advances in Neural Information Processing Systems (NIPS)*, volume 24, pages 1458–1466. MIT Press, Cambridge, MA, 2011.

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

Shai Shalev-Shwartz, Ohad Shamir, Nathan Srebro, and Karthik Sridharan. Stochastic convex optimization. In Sanjoy Dasgupta and Adam Klivans, editors, *Proc. of the 22th Annual Conference on Learning Theory (COLT'09)*, Montreal, Quebec, June 18–21 2009.

Ohad Shamir. Without-replacement sampling for stochastic gradient methods: Convergence results and application to distributed optimization. In Daniel D. Lee, Ulrike von Luxburg, and Isabelle Guyon, editors, *Advances in Neural Information Processing Systems (NIPS)*, volume 29, pages 46–54. MIT Press, Cambridge, MA, 2016.

Ohad Shamir, Nati Srebro, and Tong Zhang. Communication-efficient distributed optimization using an approximate Newton-type method. In Eric Xing and Tony Jebara, editors, *Proc. of the 31st Int. Conf. Machine Learning (ICML 2014)*, pages 1000–1008, Beijing, China, June 21–26 2014.

Vladimir N. Vapnik. *The Nature of Statistical Learning Theory*. Springer Series in Information Science and Statistics. Springer-Verlag, Berlin, second edition, 2000.

Jialei Wang, Weiran Wang, and Nathan Srebro. Memory and communication efficient distributed stochastic optimization with minibatch prox. In Satyen Kale and Ohad Shamir, editors, *Annual Conference on Learning Theory*, Amsterdam, Netherlands, July 7–10 2017.

Matthew D. Zeiler. ADADELTA: An adaptive learning rate method. arXiv:1212.5701 [cs.LG], December 2012.