ISE

Industrial and Systems Engineering

Stochastic Recursive Gradient Algorithm for Nonconvex Optimization

LAM M. NGUYEN, JIE LIU, KATYA SCHEINBERG, AND MARTIN TAKÁČ

Department of Industrial and Systems Engineering, Lehigh University, USA

ISE Technical Report 17T-007

LEHIGH UNIVERSITY

Stochastic Recursive Gradient Algorithm for Nonconvex Optimization

Lam M. Nguyen
Industrial and Systems Engineering
Lehigh University, USA
lamnguyen.mltd@gmail.com

Jie Liu
Industrial and Systems Engineering
Lehigh University, USA
jie.liu.2018@gmail.com

Katya Scheinberg
Industrial and Systems Engineering
Lehigh University, USA
On leave at University of Oxford, UK
katyas@lehigh.edu

Martin Takáč
Industrial and Systems Engineering
Lehigh University, USA
Takac.MT@gmail.com

Abstract

In this paper, we study and analyze the mini-batch version of StochAstic Recursive grAdient algoritHm (SARAH), a method employing the stochastic recursive gradient, for solving empirical loss minimization for the case of nonconvex losses. We provide a sublinear convergence rate (to stationary points) for general nonconvex functions and a linear convergence rate for gradient dominated functions, both of which have some advantages compared to other modern stochastic gradient algorithms for nonconvex losses.

1 Introduction

We are interested in the following finite-sum minimization problem

$$\min_{w\in\mathbb{R}^d}\left\{P(w)\overset{\text{def}}{=}\frac{1}{n}\sum_{i\in[n]}f_i(w)\right\}, \tag{1}$$

where each f_i, $i\in[n]\overset{\text{def}}{=}\{1,\ldots,n\}$, is smooth but can be nonconvex, and P is also not necessarily convex. Throughout the paper, we assume that there exists a global optimal solution w_* of (1); in other words, there exists a lower bound $P(w_*)$ of (1), however we do not assume the knowledge of this bound and we do not seek convergence to w_*, in general.

Problems of form (1) cover a wide range of convex and nonconvex problems including but not limited to logistic regression, multi-kernel learning, conditional random fields, neural networks, etc. In many of these applications, the number n of individual components is very large, which makes the exact computation of $P(w)$ and its derivatives and thus the use of gradient descent (GD) [17] to solve (1) expensive.

A traditional approach is to employ stochastic gradient descent (SGD) [21, 23]. Recently, a large number of improved variants of stochastic gradient algorithms have emerged, including SAG/SAGA [22, 4], MISO/FINITO [13, 5], SDCA [24], SVRG/S2GD [9, 10], SARAH [16] [1].

[1] Note that numerous modifications of stochastic gradient algorithms have been proposed, including non-uniform sampling, acceleration, repeated scheme and asynchronous parallelization. In this paper, we refrain from checking and analyzing those variants, and compare only the primary methods.

All authors were supported by NSF Grant CCF-1618717. Katya Scheinberg was partially supported by NSF Grants DMS 13-19356, CCF-1320137 and CCF-1618717.

While, nonconvex problems of the form (1) are now widely used due to the recent interest in deep neural networks, the majority of methods are designed and analyzed for the convex/strongly convex cases. Limited results have been developed for the nonconvex problems [20, 3, 2], in particular, [20, 3] introduce nonconvex SVRG, and Natasha [2] is a new algorithm but a variant of SVRG for nonconvex optimization.

In this paper we develop convergence rate analysis of a mini-batch variant SARAH for nonconvex problems of the form (1). SARAH has been introduced in [16] and shown to have a sublinear rate of convergence for general convex functions, and a linear rate of convergence for strongly convex functions. As the SVRG method, SARAH has an inner and an outer loop. It has been shown in [16] that, unlike the inner loop of SVRG, the inner loop of SARAH converges. Here we explore the properties of the inner loop of SARAH for general nonconvex functions and show that it converges at the same rate as SGD, but under weaker assumptions and with better constants in the convergence rate. We then analyze the full SARAH algorithm in the case of gradient dominated functions as a special class of nonconvex functions [18, 15, 20] for which we show linear convergence to a global minimum. We will provide the definition of a gradient dominated function in Section 3. We also note that this type of function includes the case where the objective function P is strongly convex, but the component functions f_i, $i \in [n]$, are not necessarily convex.

We now summarize the complexity results of SARAH and other existing methods for nonconvex functions in Table 1. All complexity estimates are in terms of the number of calls to the *incremental first order oracle* (IFO) defined in [1], in other words computations of $(f_i(w), \nabla f_i(w))$ for some $i \in [n]$. The iteration complexity analysis aims to bound the number of iterations $\mathcal{T}$, which is needed to guarantee that $\|\nabla P(w_{\mathcal{T}})\|^2 \leq \epsilon$. In this case we will say that $w_{\mathcal{T}}$ is an ϵ-accurate solution. However, it is common practice for stochastic gradient algorithms to obtain the bound on the number of IFOs after which the algorithm can be terminated with the guaranteed the bound on the expectation, as follows,

$$\mathbb{E}[\|\nabla P(w_{\mathcal{T}})\|^2] \leq \epsilon. \tag{2}$$

It is important to note that for the stochastic algorithms discussed here, the output $w_{\mathcal{T}}$ is not the last iterate computed by the algorithm, but a randomly selected iterate from the computed sequence.

Let us discuss the results in Table 1. The analysis of SGD in [7] in performed under the assumption that $\|\nabla f_i(\cdot)\| \leq \sigma$, for all $i \in [n]$, for some fixed constant σ. This limits the applicability of the convergence results for SGD and adds dependence on σ which can be large. In contrast, convergence rate of SVRG only requires L-Lipschitz continuity of the gradient as does the analysis of SARAH. Convergence of SVRG for general nonconvex functions is better than that of the inner loop of SARAH in terms of its dependence on ϵ, but it is worse in term of its dependence on n. In addition the bound for SVRG includes an unknown universal constant ν, whose magnitude is not clear and can be quite small. Convergence rate of the full SARAH algorithm for general nonconvex functions remains an open question. In the case of τ-gradient dominated functions, full SARAH convergence rate dominates that of the other algorithms.

Table 1: Comparisons between different algorithms for nonconvex functions.

Method	GD ([14, 20])	SGD ([7, 20])	SVRG ([20])	**SARAH**
Nonconvex	$\mathcal{O}\left(\frac{nL}{\epsilon}\right)$	$\mathcal{O}\left(\frac{L\sigma^2}{\epsilon^2}\right)$	$\mathcal{O}\left(n + \frac{n^{2/3}L}{\nu\epsilon}\right)$	$\mathcal{O}\left(n + \frac{L^2}{\epsilon^2}\right)$
τ-Gradient Dominated	$\mathcal{O}\left(nL\tau \log(\frac{1}{\epsilon})\right)$	$\mathcal{O}\left(\frac{L\tau\sigma^2}{\epsilon^2}\right)$	$\mathcal{O}\left((n + \frac{n^{2/3}L\tau}{\nu}) \log(\frac{1}{\epsilon})\right)$	$\mathcal{O}\left((n + L^2\tau^2) \log(\frac{1}{\epsilon})\right)$

Our contributions. In summary, in this paper we analyze SARAH with mini-batches for nonconvex optimization. SARAH originates from the idea of momentum SGD, SAG/SAGA, SVRG and L-BFGS and is initially proposed for convex optimization, and is now proven to be effective for minimizing finite-sum problems of general nonconvex functions. We summarize the key contributions of the paper as follows.

- We study and extend SARAH framework [16] with *mini-batches* to solving *nonconvex* loss functions, which cover the popular deep neural network problems. We are able to provide a sublinear convergence rate of the inner loop of SARAH for general nonconvex functions, under milder assumptions than that of SGD.
- Like SVRG [20], SARAH algorithm is shown to enjoy linear convergence rate for τ-gradient dominated functions–a special class of possibly nonconvex functions [18, 15].

- Similarly to SVRG, SARAH maintains a constant learning rate for nonconvex optimization, and a larger mini-batch size allows the use of a more aggressive learning rate and a smaller inner loop size.
- Finally, we present numerical results, where a practical version of SARAH, introduced in [16] is shown to be competitive on standard neural network training tasks.

2 Stochastic Recursive Gradient Algorithm

The pivotal idea of SARAH, like many existing algorithms, such as SAG, SAGA and BFGS [17], is to utilize past stochastic gradient estimates to improve convergence. In contrast with SAG, SAGA and BFGS [17], SARAH does not store past information thus significantly reducing storage cost. We present SARAH as a two-loop algorithm in Figure 1, with SARAH-IN in Figure 2 describing the inner loop.

Input: $\tilde{w}_0$, the learning rate $\eta > 0$, the batch size b and the inner loop size m.
Iterate:
for $s = 1, 2, \ldots$ **do**
  $\tilde{w}_s = \text{SARAH-IN}(\tilde{w}_{s-1}, \eta, b, m)$
end for
Output: $\tilde{w}_s$

Figure 1: Algorithm SARAH

Input: $w_0 (= \tilde{w}_{s-1})$, the learning rate $\eta > 0$, the batch size b and the inner loop size m.
Evaluate the full gradient: $v_0 = \frac{1}{n}\sum_{i=1}^{n} \nabla f_i(w_0)$
Take a gradient descent step: $w_1 = w_0 - \eta v_0$
Iterate:
for $t = 1, \ldots, m-1$ **do**
  Choose a mini-batch $I_t \subseteq [n]$ of size b uniformly at random (without replacement)
  Update the stochastic recursive gradient:

$$v_t = \frac{1}{b}\sum_{i \in I_t} [\nabla f_i(w_t) - \nabla f_i(w_{t-1})] + v_{t-1} \tag{3}$$

  Update the iterate: $w_{t+1} = w_t - \eta v_t$
end for
$\tilde{w} = w_t$ with t chosen uniformly randomly from $\{0, 1, \ldots, m\}$
Output: $\tilde{w}$

Figure 2: Algorithm SARAH within a single outer loop: SARAH-IN(w_0, η, b, m)

Similarly to SVRG, in each outer iteration, SARAH proceeds with the evaluation of a full gradient followed by an inner loop of m stochastic steps. SARAH requires one computation of the full gradient at the start of its inner loop and then proceeds by updating this gradient information using stochastic gradient estimates over m inner steps. Hence, each outer iteration corresponds to a cost of $\mathcal{O}(n + bm)$ component gradient evaluations (or IFOs). For simplicity let us consider the inner loop update for $b = 1$, as presented in [16]:

$$v_t = \nabla f_{i_t}(w_t) - \nabla f_{i_t}(w_{t-1}) + v_{t-1}, \tag{4}$$

Note that unlike SVRG, which uses the gradient updates $v_t = \nabla f_{i_t}(w_t) - \nabla f_{i_t}(w_0) + v_0$, SARAH's gradient estimate v_t iteratively includes all past stochastic gradients, however, SARAH consumes a memory of $\mathcal{O}(d)$ instead of $\mathcal{O}(nd)$ in the cases of SAG/SAGA and BFGS, because this past information is simply averaged, instead of being stored.

With either $m = 1$ or $s = 1$ and $b = n$, the algorithm SARAH recovers gradient descent (GD). We remark here that we also recover the convergence rate theoretically for GD with $s = 1$ and $b = n$ In the following section, we analyze theoretical convergence properties of SARAH when applied to nonconvex functions.

3 Convergence Analysis

First, we will introduce the sublinear convergence of SARAH-IN for general nonconvex functions. Then we present the linear convergence of SARAH over a special class of gradient dominated functions [18, 15, 20]. Before proceeding to the analysis, let us start by stating some assumptions.

Assumption 1 (L-smooth)**.** *Each $f_i : \mathbb{R}^d \to \mathbb{R}$, $i \in [n]$, is L-smooth, i.e., there exists a constant $L > 0$ such that*

$$\|\nabla f_i(w) - \nabla f_i(w')\| \leq L\|w - w'\|, \ \forall w, w' \in \mathbb{R}^d. \tag{5}$$

Assumption 1 implies that P is also L-smooth. Then, by the property of L-smooth function (in [14]), we have

$$P(w) \leq P(w') + \nabla P(w')^T(w - w') + \frac{L}{2}\|w - w'\|^2, \ \forall w, w' \in \mathbb{R}^d. \tag{6}$$

The following assumption will be made only when appropriate, otherwise, it will be dropped.

Assumption 2 (τ-gradient dominated)**.** *P is τ-gradient dominated, i.e., there exists a constant $\tau > 0$ such that $\forall w \in \mathbb{R}^d$,*

$$P(w) - P(w_*) \leq \tau\|\nabla P(w)\|^2, \tag{7}$$

where w_* is a global minimizer of P.

We can observe that every stationary point of the τ-gradient dominated function P is a global minimizer. However, such a function P needs not necessarily be convex. If P is *μ-strongly convex* (but each f_i, $i \in [n]$, is possibly nonconvex), then $2\mu[P(w) - P(w_*)] \leq \|\nabla P(w)\|^2$, $\forall w \in \mathbb{R}^d$. Thus, a μ-strongly convex function is also $1/(2\mu)$-gradient dominated.

The following two results - Lemmas 1 and 2 - are essentially the same as Lemmas 1 and 2 in [16] with a slight modification to include the case when b is not necessarily equal to 1. We present the proofs in the supplementary material for completeness.

Lemma 1. *Suppose that Assumption 1 holds. Consider SARAH-IN (SARAH within a single outer loop in Figure 2), then we have*

$$\sum_{t=0}^{m} \mathbb{E}[\|\nabla P(w_t)\|^2] \leq \frac{2}{\eta}[P(w_0) - P(w_*)] + \sum_{t=0}^{m} \mathbb{E}[\|\nabla P(w_t) - v_t\|^2] - (1 - L\eta)\sum_{t=0}^{m} \mathbb{E}[\|v_t\|^2], \tag{8}$$

where w_* is a global minimizer of P.

Lemma 2. *Suppose that Assumption 1 holds. Consider v_t defined by* (3) *in SARAH-IN, then for any $t \geq 1$,*

$$\mathbb{E}[\|\nabla P(w_t) - v_t\|^2] = \sum_{j=1}^{t} \mathbb{E}[\|v_j - v_{j-1}\|^2] - \sum_{j=1}^{t} \mathbb{E}[\|\nabla P(w_j) - \nabla P(w_{j-1})\|^2].$$

With the above Lemmas, we can derive the following upper bound for $\mathbb{E}[\|\nabla P(w_t) - v_t\|^2]$.

Lemma 3. *Suppose that Assumption 1 holds. Consider v_t defined by* (3) *in SARAH-IN. Then for any $t \geq 1$,*

$$\mathbb{E}[\|\nabla P(w_t) - v_t\|^2] \leq \frac{1}{b}\left(\frac{n-b}{n-1}\right) L^2\eta^2 \sum_{j=1}^{t} \mathbb{E}[\|v_{j-1}\|^2].$$

The proof of Lemma 3 is provided in the supplementary material. Using the above lemmas, we are now able to obtain the following convergence rate result for SARAH-IN.

Theorem 1. *Suppose that Assumption 1 holds. Consider SARAH-IN (SARAH within a single outer loop in Figure 2) with*

$$\eta \leq \frac{2}{L\left(\sqrt{1 + \frac{4m}{b}\left(\frac{n-b}{n-1}\right)} + 1\right)}. \tag{9}$$

Then we have

$$\mathbb{E}[\|\nabla P(\tilde{w})\|^2] \leq \frac{2}{\eta(m+1)}[P(w_0) - P(w_*)],$$

where w_* is a global minimizer of P, and $\tilde{w} = w_t$, where t is chosen uniformly at random from $\{0, 1, \ldots, m\}$.

This result shows a sublinear convergence rate for SARAH-IN with increasing m. Consequently, with $b = 1$ and $\eta = \frac{2}{L(\sqrt{1+4m}+1)}$, to obtain

$$\mathbb{E}[\|\nabla P(\tilde{w})\|^2] \leq \frac{L(\sqrt{1+4m}+1)}{(m+1)}[P(w_0) - P(w_*)] \leq \epsilon,$$

it is sufficient to make $m = \mathcal{O}(L^2/\epsilon^2)$. Hence, the total complexity to achieve an ϵ-accurate solution is $(n + 2m) = \mathcal{O}(n + L^2/\epsilon^2)$. Therefore, we have the following conclusion for complexity bound.

Corollary 1. *Suppose that Assumption 1 holds. Consider SARAH within a single outer iteration with batch size $b = 1$ and the learning rate $\eta = \mathcal{O}(1/(L\sqrt{m}))$ where m is the total number of iterations, then $\|\nabla P(w_t)\|^2$ converges sublinearly in expectation with a rate of $\mathcal{O}(L/\sqrt{m})$, and therefore, the total complexity to achieve an ϵ-accurate solution defined in* (2) *is $\mathcal{O}(n + L^2/\epsilon^2)$.*

Finally, we present the result for SARAH with multiple outer iterations in application to the class of gradient dominated functions defined in (7).

Theorem 2. *Suppose that Assumptions 1 and 2 hold. Consider SARAH (in Figure 1) with η and m such that*

$$\eta \leq \frac{2}{L\left(\sqrt{1 + \frac{4m}{b}\left(\frac{n-b}{n-1}\right)} + 1\right)} \quad \text{and} \quad \frac{\eta(m+1)}{2} > \tau.$$

Then we have

$$\mathbb{E}[\|\nabla P(\tilde{w}_s)\|^2] \leq (\bar{\gamma}_m)^s \|\nabla P(\tilde{w}_0)\|^2,$$

where

$$\bar{\gamma}_m = \frac{2\tau}{\eta(m+1)} < 1.$$

Consider the case when $b = 1$ and $\eta = \frac{2}{L(\sqrt{1+4m}+1)}$. We need $m = \mathcal{O}\left(L^2\tau^2\right)$ to satisfy $\frac{\eta(m+1)}{2} = \frac{m+1}{L\sqrt{1+4m}+1} > \tau$. To obtain

$$\mathbb{E}[\|\nabla P(\tilde{w}_s)\|^2] \leq (\bar{\gamma}_m)^s \|\nabla P(\tilde{w}_0)\|^2 \leq \epsilon,$$

it is sufficient to have $s = \mathcal{O}\left(\log(1/\epsilon)\right)$. This implies the total complexity to achieve an ϵ-accurate solution is $(n + 2m)s = \mathcal{O}\left((n + L^2\tau^2)\log(1/\epsilon)\right)$ and we can summarize the conclution as follows.

Corollary 2. *Suppose that Assumptions 1 and 2 hold. Consider SARAH with parameters from Theorem 2 with batch size $b = 1$ and the learning rate $\eta = \mathcal{O}\left(1/(L\sqrt{m})\right)$, then the total complexity to achieve an ϵ-accurate solution defined in* (2) *is $\mathcal{O}\left((n + L^2\tau^2)\log(1/\epsilon)\right)$.*

4 Discussions on the mini-batches sizes

Let us discuss two simple corollaries of Theorem 1.

The first corollary is obtained trivially by substituting the learning rate into the complexity bound in Theorem 1.

Corollary 3. *Suppose that Assumption 1 holds. Consider SARAH-IN (SARAH within a single outer loop in Figure 2) with*

$$\eta = \frac{2}{L\left(\sqrt{1 + \frac{4m}{b}\left(\frac{n-b}{n-1}\right)} + 1\right)}. \tag{10}$$

Then we have

$$\mathbb{E}[\|\nabla P(\tilde{w})\|^2] \leq \frac{L\left(\sqrt{1+\frac{4m}{b}\left(\frac{n-b}{n-1}\right)}+1\right)}{(m+1)}[P(w_0)-P(w_*)],$$

where w_* is a global minimizer of P, and $\tilde{w} = w_t$, where t is chosen uniformly at random from $\{0, 1, \ldots, m\}$.

Remark 1. *We can clearly observe that the rate of convergence for SARAH-IN depends on the size of b. For a larger value of b, we can use a more aggressive learning rate and it requires the smaller number of iterations to achieve an ϵ-accurate solution. In particular, when $b = n$, SARAH-IN reduces to the GD method and its convergence rate becomes that of gradient descent,*

$$\mathbb{E}[\|\nabla P(\tilde{w})\|^2] \leq \frac{2L}{(m+1)}[P(w_0)-P(w_*)],$$

and the total complexity to achieve an ϵ-accurate solution is $n \cdot m = \mathcal{O}\left(\frac{nL}{\epsilon}\right)$. However, the total work in terms of IFOs increases with b. When $b \neq n$, the total complexity to achieve an ϵ-accurate solution is $(n + 2bm) = \mathcal{O}\left(n + \frac{L^2}{\epsilon^2}\left(\frac{n-b}{n-1}\right)\right)$.

Let us set $m = n - 1$ in Corollary 3, we can achieve the following result.

Corollary 4. *Suppose that Assumption 1 holds. Consider SARAH-IN with $m = n - 1$, and*

$$\eta = \frac{2}{L\left(\sqrt{4(n/b)-3}+1\right)}.$$

Then we have

$$\mathbb{E}[\|\nabla P(\tilde{w})\|^2] \leq \frac{L\left(\sqrt{4(n/b)-3}+1\right)}{n}[P(w_0)-P(w_*)],$$

where w_* is a global minimizer of P, and $\tilde{w} = w_t$, where t is chosen uniformly at random from $\{0, 1, \ldots, n-1\}$.

Remark 2. *For SARAH-IN with the number of iterations $m = n - 1$ and the learning rate $\eta = \mathcal{O}\left(1/(L\sqrt{(n/b)})\right)$, we could achieve a convergence rate of $\mathcal{O}(L/\sqrt{bn})$. We can observe that the value of b significantly affects the rate. For example, when $b = n/\beta$, $\beta > 1$ and $b = n^\alpha$, $\alpha < 1$, the convergence rates become $\mathcal{O}(L\sqrt{\beta}/n)$ and $\mathcal{O}(L/\sqrt{n^{\alpha+1}})$, respectively.*

5 Numerical Experiments

We now turn to the numerical study and conduct experiments on the multiclass classification problem with neural networks, which is the typical challegeing nonconvex problem in machine learning.

SARAH+ as a Practical Variant [16] proposes SARAH+ as a practical variant of SARAH. Now we propose SARAH+ for the nonconvex optimization by running Algorithm SARAH (Figure 1) with the following SARAH-IN algorithm (Figure 3). Notice that SARAH+ is different from SARAH in that the inner loop is terminated adaptively instead of using a fixed choice of the inner loop size m. This is idea is based on the fact that the norm $\|v_t\|$ converges to zero expectation, which has been both proven theoretically and verified numerically for convex optimization in [16]. Under the assumption that similar behavior happens in the nonconvex case, instead of tuning the inner loop size for SARAH, we believe that a proper choice of the ratio γ below, the automatic loop termination can give superior or competitive performances.

Input: $w_0(=\tilde{w}_{s-1})$, the learning rate $\eta > 0$, the batch size b and the maximum inner loop size m.

Evaluate the full gradient: $v_0 = \frac{1}{n}\sum_{i=1}^{n} \nabla f_i(w_0)$
Take a gradient descent step: $w_1 = w_0 - \eta v_0$
Iterate:
while $\|v_{t-1}\|^2 > \gamma \|v_0\|^2$ **and** $t < m$ **do**
Choose a mini-batch $I_t \subseteq [n]$ of size b uniformly at random (without replacement)
Update the stochastic recursive gradient:

$$v_t = \frac{1}{b}\sum_{i \in I_t} [\nabla f_i(w_t) - \nabla f_i(w_{t-1})] + v_{t-1}$$

Update the iterate and index: $w_{t+1} = w_t - \eta v_t$; $t = t + 1$
end while
$\tilde{w} = w_t$ with t chosen uniformly randomly from $\{0, 1, \ldots, m\}$
Output: $\tilde{w}$

Figure 3: Algorithm SARAH within a single outer loop: SARAH-IN(w_0, η, b, m)

Networks and Datasets We perform numerical experiments with neural nets with one fully connected hidden layer of n_h nodes, followed by a fully connected output layer which feeds into the softmax regression and cross entropy objective, with the weight decay regularizer (ℓ_2-regularizer) with parameter λ. We test the performance on the datasets MNIST [12] [2] and CIFAR10 [11] [3] with $n_h = 300, \lambda =$1e-04 and $n_h = 100, \lambda =$1e-03, respectively. Both datasets have 10 classes, i.e. 10 softmax output nodes in the network, and are normalized to interval $[0, 1]$ as a simple data pre-processing. This network of MNIST achieves the best performance for neural nets with a single hidden layer. Information on both datasets is also available in Table 2.

Optimization Details We compare the efficiency of SARAH, SARAH+ [16], SVRG [20], AdaGrad [6] and SGD-M (momentum SGD [19, 25]) [4] numerically in terms of number of effective data passes, where the last two algorithms are efficient SGD variants available in the Google open-source library *Tensorflow* [5]. As the choice of initialization for the weight parameters is very important, we apply a widely used mechanism called *normalized initialization* [8] where the weight parameters between layers j and $j+1$ are sampled uniformly from $\left[-\sqrt{6/(n_j + n_{j+1})}, \sqrt{6/(n_j + n_{j+1})}\right]$. In addition, we use mini-batch size $b = 10$ in all the algorithms.

Table 2: Summary of statistics and best parameters of all the algorithms for the two datasets.

Dataset	Number of Samples (n_{train}, n_{test})	Dimensions (d)	SARAH (m^*, η^*)	SARAH+ (η^*)	SVRG (m^*, η^*)	AdaGrad (δ^*, η^*)	SGD-M (γ^*, η^*)
MNIST	(60,000, 10,000)	784	(0.1n, 0.08)	0.2	(0.4n, 0.08)	(0.01, 0.1)	(0.7, 0.01)
CIFAR10	(50,000, 10,000)	3072	(0.4n, 0.03)	0.02	(0.8n, 0.02)	(0.05, 1.0)	(0.7, 0.001)

Performance and Comparison We present the optimal choices of optimization parameters for the mentioned algorithms in Table 2, as well as their performance in Figure 4. As for the optimization parameters we consistently use the ratio 0.7 in SARAH+, while for all the others, we need to tune two parameters, including η^* for optimal learning rates, m^* for optimal inner loop size, δ^* for the optimal initial accumulator and γ^* for the optimal momentum. For the tuning of the parameters, reasonable ranges for the parameters have been scanned and we selected the best parameters in terms ofthe training error reduction.

[2] Available at `http://yann.lecun.com/exdb/mnist/`.

[3] Available at `https://www.cs.toronto.edu/~kriz/cifar.html`.

[4] While SARAH, SVRG, SGD have been proven effective for nonconvex optimization, as far as we know, the SGD variants AdaGrad and SGD-M do not have theoretical convergence for nonconvex optimization.

[5] See `https://www.tensorflow.org`.

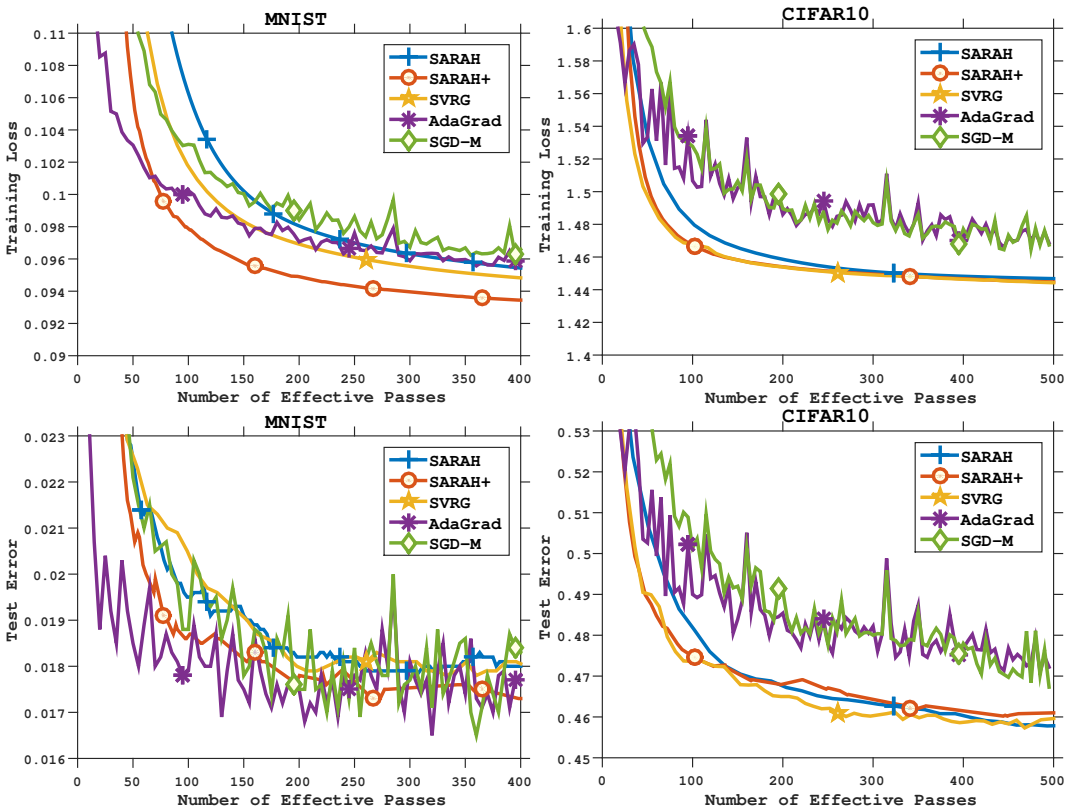

Figure 4: An example of ℓ_2-regularized neural nets on *MNIST* and *CIFAR10* training/testing datasets for SARAH, SARAH+, SVRG, AdaGrad and SGD-M.

Figure 4 compares the training losses (top) and the test errors (bottom), obtained by the tested algorithms on *MNIST* and *CIFAR10*, in terms of the number of effective passes through the data. On the *MNIST* dataset, which is deemed to be easier for traning, all the methods achieve similar performance in the end; however, SARAH(+) and SVRG stabilize faster than AdaGrad and SGD-M - the two of the most popular SGD variants; meanwhile, SARAH+ has shown superior performance in minimizing the training loss. For the other, more difficult, *CIFAR10* dataset, SARAH(+) and SVRG improve upon the training accuracy considerably in comparison with AdaGrad and SGD-M, and as a result, a similar advantage can be seen in the test error reduction.

6 Conclusion

In this paper of work, we study and extend SARAH framework to nonconvex optimization, also admitting the practical variant, SARAH+. For smooth nonconvex functions, the inner loop of SARAH achieves the best sublinear convergence rate in the literature, while the full variant of SARAH has the same linear convergence rate and same as SVRG, for a special class of gradient dominated functions. In addition, we also analyze the dependence of the convergence of SARAH on the size of the mini-batches. In the end, we validate SARAH(+) numerically in comparison with SVRG, AdaGrad and SGD-M, with the popular nonconvex application of neural networks.

References

[1] Alekh Agarwal and Leon Bottou. A lower bound for the optimization of finite sums. In *ICML*, pages 78–86, 2015.

[2] Zeyuan Allen Zhu. Natasha: Faster non-convex stochastic optimization via strongly non-convex parameter. *arXiv preprint arXiv:1702.00763*, 2017.

[3] Zeyuan Allen-Zhu and Elad Hazan. Variance reduction for faster non-convex optimization. In *ICML*, pages 699–707, 2016.

[4] Aaron Defazio, Francis Bach, and Simon Lacoste-Julien. SAGA: A fast incremental gradient method with support for non-strongly convex composite objectives. In *NIPS*, pages 1646–1654, 2014.

[5] Aaron Defazio, Justin Domke, and Tibério Caetano. A faster, permutable incremental gradient method for big data problems. In *ICML*, pages 1125–1133, 2014.

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

[7] Saeed Ghadimi and Guanghui Lan. Stochastic first- and zeroth-order methods for nonconvex stochastic programming. *SIAM Journal on Optimization*, 23(4):2341–2368, 2013.

[8] Xavier Glorot and Yoshua Bengio. Understanding the difficulty of training deep feedforward neural networks. In *AISTATS*, 2010.

[9] Rie Johnson and Tong Zhang. Accelerating stochastic gradient descent using predictive variance reduction. In *NIPS*, pages 315–323, 2013.

[10] Jakub Konečný, Jie Liu, Peter Richtárik, and Martin Takáč. Mini-batch semi-stochastic gradient descent in the proximal setting. *IEEE Journal of Selected Topics in Signal Processing*, 10:242–255, 2016.

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

[12] Yann Lecun, Léon Bottou, Yoshua Bengio, and Patrick Haffner. Gradient-based learning applied to document recognition. In *Proceedings of the IEEE*, pages 2278–2324, 1998.

[13] Julien Mairal. Optimization with first-order surrogate functions. In *ICML*, pages 783–791, 2013.

[14] Yurii Nesterov. *Introductory lectures on convex optimization : a basic course*. Applied optimization. Kluwer Academic Publ., Boston, Dordrecht, London, 2004.

[15] Yurii Nesterov and Boris T Polyak. Cubic regularization of newton method and its global performance. *Mathematical Programming*, 108(1):177–205, 2006.

[16] Lam Nguyen, Jie Liu, Katya Scheinberg, and Martin Takáč. SARAH: A novel method for machine learning problems using stochastic recursive gradient. *To appear in ICML*, 2017.

[17] Jorge Nocedal and Stephen J. Wright. *Numerical Optimization*. Springer, New York, 2nd edition, 2006.

[18] Boris T Polyak. Gradient methods for the minimisation of functionals. *USSR Computational Mathematics and Mathematical Physics*, 3(4):864–878, 1963.

[19] Boris T. Polyak. Some methods of speeding up the convergence of iteration methods. *USSR Computational Mathematics and Mathematical Physics*, 4(5):1–17, 1964.

[20] Sashank J. Reddi, Ahmed Hefny, Suvrit Sra, Barnabás Póczos, and Alexander J. Smola. Stochastic variance reduction for nonconvex optimization. In *ICML*, pages 314–323, 2016.

[21] Herbert Robbins and Sutton Monro. A stochastic approximation method. *The Annals of Mathematical Statistics*, 22(3):400–407, 1951.

[22] Mark Schmidt, Nicolas Le Roux, and Francis Bach. Minimizing finite sums with the stochastic average gradient. *Mathematical Programming*, pages 1–30, 2016.

[23] Shai Shalev-Shwartz, Yoram Singer, Nathan Srebro, and Andrew Cotter. Pegasos: Primal estimated sub-gradient solver for SVM. *Mathematical Programming*, 127(1):3–30, 2011.

[24] Shai Shalev-Shwartz and Tong Zhang. Stochastic dual coordinate ascent methods for regularized loss. *Journal of Machine Learning Research*, 14(1):567–599, 2013.

[25] Ilya Sutskever, James Martens, George Dahl, and Geoffrey Hinton. On the importance of initialization and momentum in deep learning. In *ICML*, pages 1139–1147, 2013.

Supplementary Material

A Technical Proofs

A.1 Proof of Lemma 1

By Assumption 1 and $w_{t+1} = w_t - \eta v_t$, we have

$$\begin{aligned}\mathbb{E}[P(w_{t+1})] &\overset{(6)}{\leq} \mathbb{E}[P(w_t)] - \eta\mathbb{E}[\nabla P(w_t)^T v_t] + \frac{L\eta^2}{2}\mathbb{E}[\|v_t\|^2] \\ &= \mathbb{E}[P(w_t)] - \frac{\eta}{2}\mathbb{E}[\|\nabla P(w_t)\|^2] + \frac{\eta}{2}\mathbb{E}[\|\nabla P(w_t) - v_t\|^2] - \left(\frac{\eta}{2} - \frac{L\eta^2}{2}\right)\mathbb{E}[\|v_t\|^2],\end{aligned}$$

where the last equality follows from the fact $r^T q = \frac{1}{2}\left[\|r\|^2 + \|q\|^2 - \|r-q\|^2\right]$, for any $r, q \in \mathbb{R}^d$.

By summing over $t = 0, \ldots, m$, we have

$$\mathbb{E}[P(w_{m+1})] \leq \mathbb{E}[P(w_0)] - \frac{\eta}{2}\sum_{t=0}^{m}\mathbb{E}[\|\nabla P(w_t)\|^2] + \frac{\eta}{2}\sum_{t=0}^{m}\mathbb{E}[\|\nabla P(w_t) - v_t\|^2] - \left(\frac{\eta}{2} - \frac{L\eta^2}{2}\right)\sum_{t=0}^{m}\mathbb{E}[\|v_t\|^2],$$

which is equivalent to ($\eta > 0$):

$$\begin{aligned}\sum_{t=0}^{m}\mathbb{E}[\|\nabla P(w_t)\|^2] &\leq \frac{2}{\eta}\mathbb{E}[P(w_0) - P(w_{m+1})] + \sum_{t=0}^{m}\mathbb{E}[\|\nabla P(w_t) - v_t\|^2] - (1 - L\eta)\sum_{t=0}^{m}\mathbb{E}[\|v_t\|^2] \\ &\leq \frac{2}{\eta}[P(w_0) - P(w_*)] + \sum_{t=0}^{m}\mathbb{E}[\|\nabla P(w_t) - v_t\|^2] - (1 - L\eta)\sum_{t=0}^{m}\mathbb{E}[\|v_t\|^2],\end{aligned}$$

where the last inequality follows since w_* is a global minimizer of P. (Note that w_0 is given.)

A.2 Proof of Lemma 2

Let $\mathcal{F}_j = \sigma(w_0, i_1, i_2, \ldots, i_{j-1})$ be the σ-algebra generated by $w_0, i_1, i_2, \ldots, i_{j-1}$; $\mathcal{F}_0 = \mathcal{F}_1 = \sigma(w_0)$. Note that $\mathcal{F}_j$ also contains all the information of $w_0, \ldots, w_j$ as well as $v_0, \ldots, v_{j-1}$. For $j \geq 1$, we have

$$\begin{aligned}\mathbb{E}[\|\nabla P(w_j) - v_j\|^2|\mathcal{F}_j] &= \mathbb{E}[\|[\nabla P(w_{j-1}) - v_{j-1}] + [\nabla P(w_j) - \nabla P(w_{j-1})] - [v_j - v_{j-1}]\|^2|\mathcal{F}_j] \\ &= \|\nabla P(w_{j-1}) - v_{j-1}\|^2 + \|\nabla P(w_j) - \nabla P(w_{j-1})\|^2 + \mathbb{E}[\|v_j - v_{j-1}\|^2|\mathcal{F}_j] \\ &\quad + 2(\nabla P(w_{j-1}) - v_{j-1})^T(\nabla P(w_j) - \nabla P(w_{j-1})) \\ &\quad - 2(\nabla P(w_{j-1}) - v_{j-1})^T\mathbb{E}[v_j - v_{j-1}|\mathcal{F}_j] \\ &\quad - 2(\nabla P(w_j) - \nabla P(w_{j-1}))^T\mathbb{E}[v_j - v_{j-1}|\mathcal{F}_j] \\ &= \|\nabla P(w_{j-1}) - v_{j-1}\|^2 - \|\nabla P(w_j) - \nabla P(w_{j-1})\|^2 + \mathbb{E}[\|v_j - v_{j-1}\|^2|\mathcal{F}_j],\end{aligned}$$

where the last equality follows from

$$\begin{aligned}\mathbb{E}[v_j - v_{j-1}|\mathcal{F}_j] &\overset{(3)}{=} \mathbb{E}\Big[\frac{1}{b}\sum_{i\in I_j}[\nabla f_i(w_j) - \nabla f_i(w_{j-1})]\Big|\mathcal{F}_j\Big] \\ &= \frac{1}{b}\cdot\frac{b}{n}\sum_{i=1}^{n}[\nabla f_i(w_j) - \nabla f_i(w_{j-1})] = \nabla P(w_j) - \nabla P(w_{j-1}).\end{aligned}$$

By taking expectation for the above equation, we have

$$\mathbb{E}[\|\nabla P(w_j) - v_j\|^2] = \mathbb{E}[\|\nabla P(w_{j-1}) - v_{j-1}\|^2] - \mathbb{E}[\|\nabla P(w_j) - \nabla P(w_{j-1})\|^2] + \mathbb{E}[\|v_j - v_{j-1}\|^2].$$

Note that $\|\nabla P(w_0) - v_0\|^2 = 0$. By summing over $j = 1, \ldots, t$ ($t \geq 1$), we have

$$\mathbb{E}[\|\nabla P(w_t) - v_t\|^2] = \sum_{j=1}^{t} \mathbb{E}[\|v_j - v_{j-1}\|^2] - \sum_{j=1}^{t} \mathbb{E}[\|\nabla P(w_j) - \nabla P(w_{j-1})\|^2].$$

A.3 Proof of Lemma 3

Let

$$\xi_t = \nabla f_t(w_j) - \nabla f_t(w_{j-1}). \tag{11}$$

We have

$$\begin{aligned}
&\mathbb{E}[\|v_j - v_{j-1}\|^2|\mathcal{F}_j] - \|\nabla P(w_j) - \nabla P(w_{j-1})\|^2 \\
&\overset{(3)}{=} \mathbb{E}\Big[\Big\|\frac{1}{b}\sum_{i\in I_j}[\nabla f_i(w_j) - \nabla f_i(w_{j-1})]\Big\|^2\Big|\mathcal{F}_j\Big] - \Big\|\frac{1}{n}\sum_{i=1}^{n}[\nabla f_i(w_j) - \nabla f_i(w_{j-1})]\Big\|^2 \\
&\overset{(11)}{=} \mathbb{E}\Big[\Big\|\frac{1}{b}\sum_{i\in I_j}\xi_i\Big\|^2\Big|\mathcal{F}_j\Big] - \Big\|\frac{1}{n}\sum_{i=1}^{n}\xi_i\Big\|^2 \\
&= \frac{1}{b^2}\mathbb{E}\Big[\sum_{i\in I_j}\sum_{k\in I_j}\xi_i^T\xi_k\Big|\mathcal{F}_j\Big] - \frac{1}{n^2}\sum_{i=1}^{n}\sum_{k=1}^{n}\xi_i^T\xi_k \\
&= \frac{1}{b^2}\mathbb{E}\Big[\sum_{i\neq k\in I_j}\xi_i^T\xi_k + \sum_{i\in I_j}\xi_i^T\xi_i\Big|\mathcal{F}_j\Big] - \frac{1}{n^2}\sum_{i=1}^{n}\sum_{k=1}^{n}\xi_i^T\xi_k \\
&= \frac{1}{b^2}\Big[\frac{b}{n}\frac{(b-1)}{(n-1)}\sum_{i\neq k}\xi_i^T\xi_k + \frac{b}{n}\sum_{i=1}^{n}\xi_i^T\xi_i\Big] - \frac{1}{n^2}\sum_{i=1}^{n}\sum_{k=1}^{n}\xi_i^T\xi_k \\
&= \frac{1}{b^2}\Big[\frac{b}{n}\frac{(b-1)}{(n-1)}\sum_{i=1}^{n}\sum_{k=1}^{n}\xi_i^T\xi_k + \Big(\frac{b}{n} - \frac{b}{n}\frac{(b-1)}{(n-1)}\Big)\sum_{i=1}^{n}\xi_i^T\xi_i\Big] - \frac{1}{n^2}\sum_{i=1}^{n}\sum_{k=1}^{n}\xi_i^T\xi_k \\
&= \frac{1}{bn}\Big[\Big(\frac{(b-1)}{(n-1)} - \frac{b}{n}\Big)\sum_{i=1}^{n}\sum_{k=1}^{n}\xi_i^T\xi_k + \frac{(n-b)}{(n-1)}\sum_{i=1}^{n}\xi_i^T\xi_i\Big] \\
&= \frac{1}{bn}\Big(\frac{n-b}{n-1}\Big)\Big[-\frac{1}{n}\sum_{i=1}^{n}\sum_{k=1}^{n}\xi_i^T\xi_k + \sum_{i=1}^{n}\xi_i^T\xi_i\Big] \\
&= \frac{1}{bn}\Big(\frac{n-b}{n-1}\Big)\Big[-n\Big\|\frac{1}{n}\sum_{i=1}^{n}\xi_i\Big\|^2 + \sum_{i=1}^{n}\|\xi_i\|^2\Big] \\
&\leq \frac{1}{b}\Big(\frac{n-b}{n-1}\Big)\frac{1}{n}\sum_{i=1}^{n}\|\xi_i\|^2 \\
&\overset{(11)}{=} \frac{1}{b}\Big(\frac{n-b}{n-1}\Big)\frac{1}{n}\sum_{i=1}^{n}\|\nabla f_i(w_j) - \nabla f_i(w_{j-1})\|^2 \\
&\overset{(5)}{\leq} \frac{1}{b}\Big(\frac{n-b}{n-1}\Big)L^2\eta^2\frac{1}{n}\sum_{i=1}^{n}\|v_{j-1}\|^2 \\
&= \frac{1}{b}\Big(\frac{n-b}{n-1}\Big)L^2\eta^2\|v_{j-1}\|^2.
\end{aligned}$$

Hence, by taking expectation, we have

$$\mathbb{E}[\|v_j - v_{j-1}\|^2] - \mathbb{E}[\|\nabla P(w_j) - \nabla P(w_{j-1})\|^2] \leq \frac{1}{b}\left(\frac{n-b}{n-1}\right) L^2\eta^2 \mathbb{E}[\|v_{j-1}\|^2].$$

By Lemma 2, for $t \geq 1$,

$$\begin{aligned}\mathbb{E}[\|\nabla P(w_t) - v_t\|^2] &= \sum_{j=1}^{t} \mathbb{E}[\|v_j - v_{j-1}\|^2] - \sum_{j=1}^{t} \mathbb{E}[\|\nabla P(w_j) - \nabla P(w_{j-1})\|^2] \\ &\leq \frac{1}{b}\left(\frac{n-b}{n-1}\right) L^2\eta^2 \sum_{j=1}^{t} \mathbb{E}[\|v_{j-1}\|^2].\end{aligned}$$

This completes the proof.

However, the result simply follows for the case when $b = 1$ by the alternative proof. We have

$$\|v_t - v_{t-1}\|^2 \overset{(4)}{=} \|\nabla f_{i_t}(w_t) - \nabla f_{i_t}(w_{t-1})\|^2 \overset{(5)}{\leq} L^2\|w_t - w_{t-1}\|^2 = L^2\eta^2\|v_{t-1}\|^2,\ t \geq 1. \quad (12)$$

Hence, by Lemma 2, we have

$$\mathbb{E}[\|\nabla P(w_t) - v_t\|^2] \leq \sum_{j=1}^{t} \mathbb{E}[\|v_j - v_{j-1}\|^2] \overset{(12)}{\leq} L^2\eta^2 \sum_{j=1}^{t} \mathbb{E}[\|v_{j-1}\|^2].$$

A.4 Proof of Theorem 1

By Lemma 3, we have

$$\mathbb{E}[\|\nabla P(w_t) - v_t\|^2] \leq \frac{1}{b}\left(\frac{n-b}{n-1}\right) L^2\eta^2 \sum_{j=1}^{t} \mathbb{E}[\|v_{j-1}\|^2].$$

Note that $\|\nabla P(w_0) - v_0\|^2 = 0$. Hence, by summing over $t = 0, \ldots, m$ ($m \geq 1$), we have

$$\sum_{t=0}^{m} \mathbb{E}\|v_t - \nabla P(w_t)\|^2 \leq \frac{1}{b}\left(\frac{n-b}{n-1}\right) L^2\eta^2 \Big[m\mathbb{E}\|v_0\|^2 + (m-1)\mathbb{E}\|v_1\|^2 + \cdots + \mathbb{E}\|v_{m-1}\|^2\Big].$$

We have

$$\begin{aligned}&\sum_{t=0}^{m} \mathbb{E}[\|\nabla P(w_t) - v_t\|^2] - (1 - L\eta) \sum_{t=0}^{m} \mathbb{E}[\|v_t\|^2] \\ &\leq \frac{1}{b}\left(\frac{n-b}{n-1}\right) L^2\eta^2 \Big[m\mathbb{E}\|v_0\|^2 + (m-1)\mathbb{E}\|v_1\|^2 + \cdots + \mathbb{E}\|v_{m-1}\|^2\Big] \\ &- (1 - L\eta)\Big[\mathbb{E}\|v_0\|^2 + \mathbb{E}\|v_1\|^2 + \cdots + \mathbb{E}\|v_m\|^2\Big] \\ &\leq \Big[\frac{1}{b}\left(\frac{n-b}{n-1}\right) L^2\eta^2 m - (1 - L\eta)\Big] \sum_{t=1}^{m} \mathbb{E}[\|v_{t-1}\|^2] \overset{(9)}{\leq} 0, \end{aligned} \quad (13)$$

since

$$\eta = \frac{2}{L\left(\sqrt{1 + \frac{4m}{b}\left(\frac{n-b}{n-1}\right)} + 1\right)}$$

is a root of equation

$$\frac{1}{b}\left(\frac{n-b}{n-1}\right) L^2\eta^2 m - (1 - L\eta) = 0.$$

Therefore, by Lemma 1, we have

$$\sum_{t=0}^{m}\mathbb{E}[\|\nabla P(w_t)\|^2] \leq \frac{2}{\eta}[P(w_0)-P(w_*)] + \sum_{t=0}^{m}\mathbb{E}[\|\nabla P(w_t)-v_t\|^2] - (1-L\eta)\sum_{t=0}^{m}\mathbb{E}[\|v_t\|^2]$$
$$\overset{(13)}{\leq} \frac{2}{\eta}[P(w_0)-P(w_*)].$$

If $\tilde{w}=w_t$, where t is chosen uniformly at random from $\{0,1,\ldots,m\}$, then

$$\mathbb{E}[\|\nabla P(\tilde{w})\|^2] = \frac{1}{m+1}\sum_{t=0}^{m}\mathbb{E}[\|\nabla P(w_t)\|^2] \leq \frac{2}{\eta(m+1)}[P(w_0)-P(w_*)].$$

A.5 Proof of Theorem 2

Note that $\tilde{w}_s=\tilde{w}$ and $w_0=\tilde{w}_{s-1}$, $s\geq 1$. By Theorem 1, we have

$$\mathbb{E}[\|\nabla P(\tilde{w}_s)\|^2|\tilde{w}_{s-1}] = \mathbb{E}[\|\nabla P(\tilde{w})\|^2|w_0] \leq \frac{2}{\eta(m+1)}[P(w_0)-P(w_*)]$$
$$\overset{(7)}{\leq} \frac{2\tau}{\eta(m+1)}\|\nabla P(w_0)\|^2$$
$$= \frac{2\tau}{\eta(m+1)}\|\nabla P(\tilde{w}_{s-1})\|^2.$$

Hence, taking expectation to have

$$\mathbb{E}[\|\nabla P(\tilde{w}_s)\|^2] \leq \frac{2\tau}{\eta(m+1)}\mathbb{E}[\|\nabla P(\tilde{w}_{s-1})\|^2] \leq \left[\frac{2\tau}{\eta(m+1)}\right]^s \|\nabla P(\tilde{w}_0)\|^2.$$

B Additional Plots

Performance on *MNIST* with $b=20$ In addition to the plots in Section 5 with mini-batch size $b=10$, we also experimented with $b=20$ for *MNIST* where the other settings of the network remain the same. Similar trend can be observed for the algorithms, where SARAH+ seems to be the best in terms of sub-optimality/training loss while SVRG and SARAH follows with roughly comparable performance. The two SGD variants–AdaGrad and SGD-M exhibit a little worse performance.

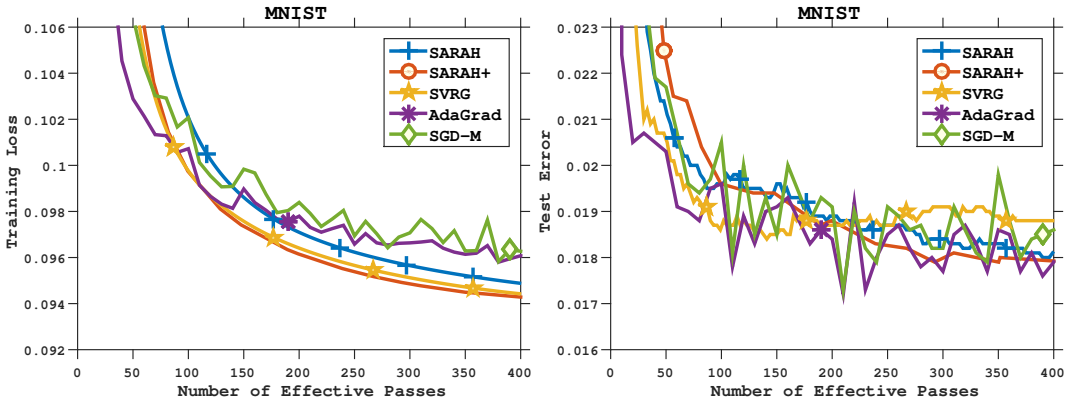

Figure 5: An example of ℓ_2-regularized neural nets on *MNIST* training dataset for SARAH, SARAH+, SVRG, AdaGrad and SGD-M with mini-batch size $b=20$.

Sensitivity of Inner Loop Size on *CIFAR10* To validate the necessity of SARAH+, we performance sensitivity analyses for the inner loop size m, that is we show the importance of the choice of

this m for SARAH and SVRG on *CIFAR10* in Figure 6, where m^*s denote the best choices what are presented in Table 2.

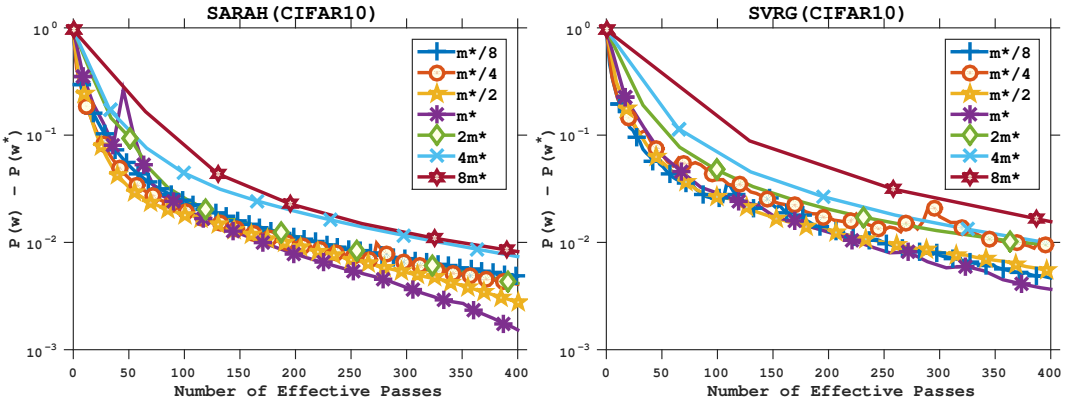

Figure 6: Sensitivity analysis of the inner loop size m for SARAH and SVRG with the example of ℓ_2-regularized neural nets on *CIFAR10* training dataset.