COST-EFFICIENT SVRG WITH ARBITRARY SAMPLING

Anonymous authors
Paper under double-blind review

ABSTRACT

We consider the problem of distributed optimization over a network, using a stochastic variance reduced gradient (SVRG) algorithm, where executing every iteration of the algorithm requires computation and exchange of gradients among network nodes. These tasks always consume network resources, including communication bandwidth and battery power, which we model as a general cost function. In this paper, we consider a modified SVRG algorithm with arbitrary sampling (SVRG-AS+), where the nodes are sampled according to some distribution. We characterize the convergence of SVRG-AS+, in terms of this distribution. We determine the distribution that minimizes the costs associated with running the algorithm, with provable convergence guarantees. We show that our approach can substantially outperform vanilla SVRG and its variants in terms of both convergence rate and total cost of running the algorithm. We then show how our approach can optimize the mini-batch size to address the tradeoff between low communication cost and fast convergence rate. Comprehensive theoretical and numerical analyses on real datasets reveal that our algorithm can significantly reduce the cost, especially in large and heterogeneous networks. Our results provide important practical insights for using machine learning over Internet-of-Things.

1 INTRODUCTION

Consider the problem of minimizing a sum of differentiable functions \( \{ f_i : \mathbb{R}^d \mapsto \mathbb{R} \}_{i \in [N]} \), with corresponding gradients \( \{ g_i : \mathbb{R}^d \mapsto \mathbb{R}^d \}_{i \in [N]} \):

\[
\mathbf{w}^* = \min_{\mathbf{w} \in \mathbb{R}^d} f(\mathbf{w}) = \min_{\mathbf{w} \in \mathbb{R}^d} \frac{1}{N} \sum_{i \in [N]} f_i(\mathbf{w}).
\]

(1)

Such problems frequently arise in statistical learning, in which each \( f_i \) could represent a regularized loss over some sampled data points. In practice, such problems are often solved using a gradient-based algorithm. Due to the large scale of many applications, most modern machine-learning approaches distribute the tasks of finding the \( N \) gradients to some computational nodes (also called workers) \( \text{[Bottou et al., 2018]} \), to enable parallel computations, or simply because the data is not available at a single place. That is, at iteration \( k \), a subset of the workers compute and send their gradients \( \{ g_i(\mathbf{w}_k) \}_i \) to a central controller (also called the master node), which updates the model and broadcasts the updated parameter \( \mathbf{w}_{k+1} \) to the workers. One of the most successful class of methods to solve (1), is the classical stochastic gradient descent and its variance-reduced extensions, including stochastic variance-reduced gradient (SVRG) and stochastic average gradient (SAGA) \( \text{[Bottou et al., 2018]; Johnson and Zhang, 2013; Defazio et al., 2014]} \). In this paper, we focus on SVRG.

In a distributed computation setting, running each iteration of the algorithm involves some costs \( c_i \), which could correspond to the number of bits (or energy or latency) needed to send \( \{ g_i(\mathbf{w}_k) \}_i \), or the computational resources needed to compute \( \{ g_i(\mathbf{w}_k) \}_i \). These costs become of paramount importance when we implement machine learning and distributed optimization algorithms on bandwidth and battery-limited wireless networks. In such networks, tight requirements on low end-to-end latency (in autonomous driving), low energy usage (Internet-of-Things), and high reliability (remote industrial operation) may render the ultimate solution, and consequently the distributed algorithm, useless \( \text{[Jeschke et al., 2017]} \). Our literature review in Section 2 shows that the existing distributed optimization solutions often ignore these important cost terms. In the case of SVRG, the gradient sampling ignores the heterogeneous costs of obtaining \( g_i(\mathbf{w}) \), for different \( i \), as well as the importance of this gradient for the convergence rate of the algorithm. Here, we address this open research problem.
In this paper, we build on SVRG with arbitrary sampling (SVRG-AS+), introduced in (Horváth and Richtarik, 2019) where the gradient sampling follows a generic multinomial distribution. Our algorithm, SVRG-AS+, allows for a variable inner loop length, reducing the amount of computations/communications of gradients in the inner loop by half, on average, compared to vanilla SVRG. We show that, when each \( f_i \) is strongly convex and \( L_i \)-smooth, the convergence rate of SVRG-AS+ is a function of \( L := \sum_i L_i/N \) instead of \( L_{\text{max}} := \max_i L_i \) in the vanilla SVRG (Johnson and Zhang, 2013). Similar results have been proved for SVRG and SAGA in smooth but nonconvex setting (Horváth and Richtarik, 2019) and for SAGA in convex setting (Qian et al., 2019). We then use our novel convergence bounds to design a minimum-cost SVRG-AS+ algorithm and transform the resulting optimization problem into a linear program. Comprehensive theoretical and numerical analyses on real datasets reveal that the optimal sampling rate of \( g_i \) is a function of \( L_i \). We then consider cost functions that model two important use cases: 1) stragglers in the federated learning case and 2) congestion in wireless communications. In both cases, we show that our minimum cost SVRG-AS+ can significantly outperform the vanilla SVRG and its state-of-the-art variants, including importance sampling ones, in terms of both total costs of running the algorithm and/or convergence rate. In particular, we show that the optimal mini-batch size depends not only on the computational loads and the number of gradient exchanges but also heavily on the communication protocol.

**Notation:** Normal font \( w \) or \( W \), bold font lowercase \( w \), bold-font capital letter \( W \), and calligraphic font \( \mathcal{W} \) denote scalar, vector, matrix, and distribution function, respectively. We let \([N] = \{1, 2, \ldots, N\}\) for any integer \( N \). We denote by \( \| \cdot \| \) the \( l_2 \) norm, by \( x^T \) the transpose of \( x \), and by \( \mathbb{1}_x \) the indicator function taking 1 when condition \( x \) holds. For easier reference, we have provided a table of notations in the appendix, where we also present proofs and extra discussions.

### 2 Literature review

**Communication-efficient distributed optimization.** Cost-efficient distributed optimization is addressed in the literature only via the notion of communication-efficiency. Example settings include networked control (Hespanha et al., 2007), distributed optimization (Tsitsiklis and Luo, 1987; Rabbat and Nowak, 2005; Zhang et al., 2012, 2015; Wang and Joshi, 2018), and machine learning (Balcan et al., 2012; Zhang et al., 2013; Jordan et al., 2018; Zhu and Lafferty, 2018; Stich et al., 2018; Karimireddy et al., 2019).

In the literature, there are two classes of approaches relevant to this paper: a) quantization of the parameter and gradient vectors at every iteration, and b) eliminating some communications at every step (Tang et al., 2020). The first category includes approaches that reduce the number of bits used to represent \( w_k \) and \( g_i(w_k) \), thereby alleviating the communication between the master node and the workers at every iteration. Recent studies have shown that proper quantization approaches can maintain the convergence to the true minimizer, as well as the convergence rate (Bernstein et al., 2018; Kamilov et al., 2018; De Sa et al., 2018; Stich et al., 2018; Magnússon et al., 2019; Karimireddy et al., 2019).

The second category includes algorithms that eliminate communication between some of the workers and the master node in some iterations (Chen et al., 2018). Chen et al. (Chen et al., 2018) proposed lazily aggregated gradient (LAG) for communication-efficient distributed learning in master-worker architectures. In LAG, each worker reports its gradient vector to the master node only if the changes to the gradient from the previous step, measured by \( l_2 \) norm, is large enough. That way, some nodes may skip sending their gradients at some iterations, which saves communication resources. Sun et al. (Sun et al., 2019) extended LAG by sending quantized gradient vectors, instead of the true values.

To the best of our knowledge, all existing works assess the convergence in terms of the number of iterations, bits transmitted, or gradients exchanged to achieve a certain solution accuracy. However, when solving a machine learning problem over a network, the main design objectives are usually latency, total energy usage, and reliability, rather than the number of algorithm iterations or bits involved. For instance, in the presence of a congested network, where sending more packets leads to more communication failures and delays, we may need a fundamental redesign of the distributed optimization algorithm to control the number of active workers based on the network conditions, rather than the gradient norm. This paper addresses cost-aware distributed optimization.
**Arbitrary and importance sampling strategies.** There has been a recent wave of works on importance and arbitrary samplings for various stochastic algorithms. Using the primal-dual gap as a measure, importance sampling has been successfully developed for randomized coordinate descent algorithms to replace the inefficient random coordinate selection of the updates (Nesterov, 2012; Allen-Zhu et al., 2016; Perekrestenko et al., 2017; Konecny et al., 2017). Stich et al. (Stich et al., 2017) extended these results to adaptive importance sampling for coordinate descent, where the sampling probability changes over time to cope with the local geometry of the optimization landscape. Gower et al. (Gower et al., 2018) introduced a class of variance reduction algorithms based on Jacobian sketching (JacSketch) in every step and developed importance sampling for SAGA in the strongly convex case. Qian et al. (Qian et al., 2019) analyzed SAGA with arbitrary sampling in the non-smooth setting. Horvath et al. (Horváth and Richtarik, 2019) analyzed the importance mini-batch sampling for SVRG and SAGA in the nonconvex setting, and Gazagnadou et al. (Gazagnadou et al., 2019) used the JacSketch algorithm to find the optimal mini-batch size for SAGA in the strongly convex and smooth setting.

Most existing theoretical results suggest that a mini-batch of size 1 gives the best solution, disagreeing with practical implementations, where much faster convergence can often be achieved using larger mini-batch sizes. However, Sebbouh et al. (Sebbouh et al., 2019) established optimal batch sizes for SVRG, showing that larger batch sizes can in fact reduce total complexity (number of iterations required to reach target accuracy). Gazagnadou et al. (Gazagnadou et al., 2019) showed, both theoretically and experimentally, that SAGA may benefit from a larger mini-batch size. In this paper, we extend those results for SVRG and show that the optimal mini-batch size depends on, not only the smoothness and strong-convexity parameters of each $f_i$, but also the communication link between the workers and the master node.

### 3 SVRG-AS+ AND CONVERGENCE RESULTS

In this section, we present our main algorithm and analyze its performance in two scenarios: running the inner loop of SVRG with either a single gradient or a mini-batch.

#### 3.1 SVRG-AS+

At the beginning of each inner loop of SVRG (also called epoch), which then runs for $T$ iterations, the master node broadcasts the parameter $\tilde{w}_k$ to the workers. At each inner iteration $t$, the master node broadcasts $w_{k,t-1}$, realizes the random variable $\xi := \xi_{k,t-1} \in [N]$, and receives $g_i(w_{k,t-1})$ from the randomly chosen worker $\xi$ (Johnson and Zhang, 2013). At the end of the inner loop, the master node picks a random iterate $\zeta$ between 1 and $T$, sets $w_k$ to $w_{k,\zeta-1}$, and updates $g_k$ for the next epoch.

In modified SVRG with arbitrary sampling (SVRG-AS+), each $\xi$ is an i.i.d. random variable with stationary multinomial distribution $P := P(p_1, p_2, \ldots, p_N)$, with $p_j := P(\xi = j)$. Due to this non-uniform sampling, we update based on a scaled version of the gradient, $h_{\xi}(w) := g_{\xi}(w)/Np_{\xi}$, rather than $g_{\xi}(w)$. The SVRG-AS+ algorithm is illustrated in Algorithm 1. Vanilla SVRG then corresponds to $p_j = 1/N$ for all $j \in [N]$. We should emphasize that our SVRG-AS+ allows a variable inner loop length (due to Lines 5 and 6) and computes only the necessary (first $\zeta$) iterates of the inner loop, as opposed to vanilla SVRG and previous SVRG-AS (Horváth and Richtarik, 2019) where such selection was at the end of the inner loop, leading to extra unnecessary computations/communications of gradients and parameter updates. This change reduces the number of inner loop iterations by half, on average, compared to vanilla SVRG, without affecting the convergence rate.

For the sake of mathematical analysis, we limit the class of objective functions to be strongly convex and smooth, though our approach may be applicable to invex (Karimi et al., 2016) and multi-convex (Xu and Yin, 2013) structures (like a deep neural network training optimization problem).

**Assumption 1.** We assume that $f(w)$ is $\mu$-strongly convex and that each gradient $g_i$ is $L_i$-Lipschitz for all $i \in [N]$. Namely, $(g_i(v) - g_i(w))^T(v - w) \geq \mu\|v - w\|^2$ and $\|g_i(v) - g_i(w)\| \leq L_i\|v - w\|$ for all $i \in [N]$ and $v$ and $w$ where $g := \sum_{i \in [N]} g_i/N$.

Next, we characterize the convergence behavior of SVRG-AS+, given in Algorithm 1. The starting point will be the following lemma, which is based on (Gazagnadou et al., 2019) Definition 2):
The solution is
We only need to track the local smoothness of the local functions, and sample according to the
An effective approach for reducing the variance of the gradient error, is to use mini-batching in
As shown in the following proposition, the step size, and consequently the convergence rate is a
Assuming that each of the functions $f_i$ is $L_i$-smooth, we can set $L = L_{\max} := \max_i L_i$ and Lemma 3 follows from [Johnson and Zhang, 2013]. Here, we extend [Johnson and Zhang, 2013] and show that the convergence is a function of expected smoothness, which can be significantly smaller than $L_{\max}$. As a result, we may use a much larger step size to substantially improve the convergence rate.

**Lemma 2.** Suppose that each of the functions $f_i$ is $L_i$-smooth for all $i \in [N]$. Then $L$ in Lemma 1 respects $L \leq \max_{i \in [N]} \{ L_i / N \}$.

**Proposition 1.** Minimizing the upper bound of the expected smoothness $L$ yields the constrained problem

$$
\min_p \max_{i \in [N]} \left\{ \frac{L_i}{N p_i} \right\} \text{ subject to } \sum_{i \in [N]} p_i = 1.
$$

The solution is $p_i^\ast = L_i / (N \bar{L})$, and the optimal $L$ is $\bar{L}$, where $\bar{L} := \sum_i L_i / N$ is the average of $L_i$’s.

As shown in the following proposition, the step size, and consequently the convergence rate is a function of $L$. Non-uniform sampling can potentially lead to a faster convergence rate than uniform sampling with $L = L_{\max}$, since $\sum_i L_i / N \leq L_{\max}$. The gain would be more prominent as $N$ increases, unless all $L_i$’s are equal. The latter is often not the case in practice, when the data are non-i.i.d. and the network nodes have their own private datasets (Li et al., 2019). Moreover, Proposition 1 implies that we can adaptively change sampling policy based on local geometry. We only need to track the local smoothness of the local functions, and sample according to the probabilities $\{ p_i^\ast = L_i / \sum_i L_i \}$ at the point $\bar{w}_k$. In the following, however, we assume that vector $p = [p_1, p_2, \ldots, p_N]^T$ is fixed for all iterations. Now, we can characterize the convergence of the SVRG-AS+ algorithm.

**Proposition 2.** Let $\alpha_k < 1 / 4 L$ and $T > 1 / (\mu c \alpha_k (1 - 4 L \alpha_k))$, and set $\Delta_k := \mathbb{E} [f(\bar{w}_k)] - f(w^\ast)$. The iterates of Algorithm 1 satisfy for any $k \in [0, K - 1]$

$$
\Delta_{k+1} \leq \sigma_k \Delta_k, \quad 0 < \sigma_k = \frac{1 + 2 L \alpha_k}{1 - 2 L \alpha_k} < 1.
$$

Proposition 2 suggests that the convergence of SVRG-AS+ depends heavily on the expected smoothness and therefore on the sampling probability vector $p$.

### 3.2 Mini-batch SVRG-AS+

An effective approach for reducing the variance of the gradient error, is to use mini-batching in the inner loop of SVRG-AS+ (Bottou et al., 2018). That is, letting $\xi$ be a random mini-batch;
see Appendix A for a formal definition. Similar to (Horváth and Richtarik 2019), we consider a stochastic definition of mini-batch size in the sense that $E[\xi] = b$. Although different from the traditional deterministic definition of mini-batch size, i.e., $\xi = b$, this new stochastic model allows for a distributed implementation of mini-batch SVRG-AS+.

The mini-batch SVRG-AS+ is almost identical to Algorithm 1 except Lines 7 and 8. Line 7 should be changed to “Every worker $i$ with probability $p_i$, independent of other workers, computes $g_i(w)$, and sends $h_i(w) := g_i(w)/Np_i$ to the master node, for both $w = w_{k,t-1}$ and $w = \bar{w}_k$. Moreover, $\mathbb{1}_{i\in\xi}$ in Line 8 should be changed to $\mathbb{1}_{i\in\xi}$ by redefining $\xi$ to be the set of sampled gradients, i.e., $\xi = \{i \mid h_i(w_{k,t-1})$ is sampled}. We have presented this algorithm in the Appendix.

**Remark 1.** Convergence of mini-batch SVRG-AS+ is the same as of Proposition 2 with the same definition for $L$ as of Lemma 2. The only difference is that $\sum_i p_i = b$ instead of being 1 in Propositions 1 and 2.

Next, we show how to use these convergence bounds to optimize the operation of SVRG-AS+ on a network with limited communication resources. Hereafter, we assume $\alpha_k = \alpha, k \in [K]$ in the following for notational simplicity.

## 4 Minimum-Cost SVRG-AS+

Here, we design a minimum cost SVRG-AS+ whose performance is at least equal to that of SVRG.

### 4.1 Outperforming Vanilla SVRG

Let $c_i$ be non-negative real numbers representing the cost of collecting the corresponding gradient $g_i(w)$ for any $w$, and $C_k$ be the cost of running iteration $k$. Assume that $\alpha$ and $T$, satisfying the conditions of Proposition 2, are given. We can formulate cost-efficient SVRG-AS+ as

\[
\begin{align*}
\text{minimize} \quad & \mathbb{E}_{\xi \sim P} [C_k] = T \sum_{i \in [N]} c_i p_i, \\
\text{subject to} \quad & \sum_{i \in [N]} p_i = 1, \quad p_i \geq 0 \quad \forall i \in [N] \\
& \alpha \leq \frac{1}{4 \max_i \{L_i / Np_i\}}, \\
& \frac{1}{\mu} \alpha' + \frac{2\alpha'}{\max_i \{L_i / Np_i\}} \leq \frac{1}{1 - 2\alpha' L_{\max}} \quad \forall \alpha' \in [0, 1/4L_{\max}],
\end{align*}
\]

where the objective function is the average sampling cost, and constraint (3d) ensures that the convergence rate of SVRG-AS+ is as good as that of SVRG with uniform sampling (i.e., $L = L_{\max}$) for any admissible step-size $\alpha'$. As shown in Appendix B, $\alpha \leq 1/4L$, where $L := \sum_{i=1}^N L_i / N$, is a sufficient condition for the feasibility of (3) at the supplementary materials. Let $j$ be any index satisfying $c_j = \min_{i \in [N]} c_i$. A solution to optimization problem (3) is then given by

\[
\begin{align*}
p_i^* &= \begin{cases} 
1 - \frac{\sum_{i \in [N] \setminus \{j\}} 4L_i}{N} \max \left\{ \alpha, \frac{1}{4L_{\max}} \right\}, & \text{if } i = j, \\
\frac{4L_j}{N} \max \left\{ \alpha, \frac{1}{4L_{\max}} \right\}, & \text{otherwise},
\end{cases}
\end{align*}
\]

and such a sequence $(p_i^*)$ exists when $\alpha \leq 1/4L$. The solution implies that except the node with minimum sampling cost $c_{\min}$, we sample at a rate that linearly depends on the smoothness parameter, $L_i$. Namely, SVRG-AS+ prefers taking fewer samples from nodes with smaller $L_i$.

Notice that we can easily change optimization problems (3) and (A.7) to find a sampling strategy that ensures a certain contraction for SVRG-AS+, namely $\sigma_k \leq \sigma_{\max}$ for some desired $\sigma_{\max}$. If the resulting problem are feasible, namely there exists a sampling strategy for which $\sigma_k \leq \sigma_{\max}$, the solution would be similar to (4). We further study this case in Section 4.2.

Moreover, we should point out that $T$ and $\alpha$ are given constants to this optimization problem. Corollary 1 in Appendix shows the interplay among $\sigma_{\max}$, $T$, and $\alpha$. Generally speaking, a smaller
\( \sigma_{\text{max}} \) (faster convergence) implies a smaller \( \alpha \), and consequently a larger \( T \). This leads to a new tradeoff in the objective function, as a smaller \( \alpha \) may lead to a smaller \( p_i \), for \( i \neq j \), and therefore smaller \( \sum c_ip_i \), but also a larger \( T \). We can address this tradeoff by optimizing over step-size, which we leave as our future work. It is also worth mentioning that our experiments show that the bounds on \( T \) and \( \alpha \) for SVRG-AS+ as well as the vanilla SVRG may be conservative in general. That is, we can violate the inequalities of Corollary 1 by using a larger \( \alpha \), and a smaller \( T \), and still converge to the optimal solution. This observation suggests that \( T \) and \( \alpha \) may be optimized as hyper-parameters of the algorithm, independent of \( p \).

### 4.2 Use Cases

**Stragglers.** A major disadvantage of Algorithm 1 is the waiting time for slow devices (i.e., stragglers or stale workers). This problem is prominent in ML over wireless networks, due to the hardware constraints and unreliability of some wireless links. For example, a node with low battery power may automatically enter energy-saving mode and drastically reduce its processing and communication resources, affecting the convergence of distributed optimization (Zhang and Simeone, 2019).

To model stragglers, we assign a high cost \( c_i \) to some worker nodes \( i \), called stragglers, while keeping the rest at a much lower level. Here, we consider SVRG-AS+ of Algorithm 1 and focus on the mini-batch SVRG-AS+ in the next use case. Referring to the optimization problem in (3), we obtain the optimal sampling probability given by (4). In particular, the solution keeps sampling from stragglers at a minimal rate, whose value depends on the smoothness \( L_i \) for their private dataset. To further improve the robustness to straggler, we may complement our importance sampling with other approaches, like data duplication (Zhang and Simeone, 2019), or asynchronous updates (Xie et al., 2019). We numerically investigate the impact of the straggler nodes on vanilla SVRG and our cost-efficient SVRG-AS+ in Section 5.

**Congestion in wireless communications.** In many cases of machine learning over networks, information exchanges happen through a common wireless channel that is shared among all workers. ALOHA and carrier-sense multiple access (CSMA) are important classes of algorithms that regulate how various workers should access the channel and send their data (gradient vectors in this case) without explicit coordination among themselves (Bertsekas et al., 2004). These algorithms are the foundations for connectivity of most modern distributed wireless systems, including Bluetooth and WiFi (Bertsekas et al., 2004).

As we have shown in Appendix C, our minimum latency SVRG-AS+ problem to ensure \( \Delta_k \leq \epsilon_1 \) for some constant \( \epsilon_1 > 0 \) reads

\[
\begin{align*}
\text{minimize} & \quad KT - \exp \left\{ \sum_{i \in [N]} p_i / r_1 \right\}, \quad \text{s.t.} \quad p_i \in \left[ \frac{2L_i}{N} \max \left\{ \frac{\alpha}{\epsilon_2}, 1, 2L_{\text{max}} \right\}, 1 \right], \quad \forall i \in [N] \\
\text{where} & \quad \epsilon_2 = \left( \frac{(\epsilon_1 / \Delta_0)^{1/K}}{1 + (\epsilon_1 / \Delta_0)^{1/K}} \right) \left( 1 + \frac{1}{\mu T \alpha} \right) - \frac{1}{\mu T \alpha}.
\end{align*}
\]

Ignoring the constraints, the optimal solution is \( \sum_i p_i = r_1 \) with the objective of \( 2.72KT / r_0 r_1 \). Moreover, the objective is quasi-convex for positive \( \sum_i p_i \) and therefore the closer to the optimal point the better objective. When \( r_1 > N \), the optimal solution is \( p_i = 1 \) for all \( i \), namely all of the nodes should transmit. In other words, the channel capacity is large enough for all workers to simultaneously report their gradient vectors with manageable cost. However, when \( r_1 < N \), we need to control the channel congestion by asking some workers to use smaller (yet feasible) \( p_i \) such that \( \sum_i p_i = r_1 \). If \( r_1 \) is too small, the optimal solution may become choosing the lower bound for all \( p_i \), leading to an even smaller mini-batch for every iteration.

Our novel cost-efficient optimization problem (5) suggests that higher transmission probabilities and consequently larger mini-batch sizes \( \sum_i p_i \) may not necessarily be optimal, even if we ignore the higher computational costs involved in obtaining extra gradients. To the best of our knowledge, this fundamental design insight has never been properly formulated in the literature.
5 EXPERIMENTAL RESULTS

Settings. In this section, we numerically characterize the convergence of the SVRG-AS+ algorithm on some real-world dataset and communication channels. We use the MNIST dataset, which has 60,000 training samples of dimension $d = 784$ and 10 classes corresponding to hand-written digits as well as the CIFAR10 dataset. We split each dataset into $N$ disjoint subsets of size $\{M_i\}_{i \in [N]}$, and assume each node $i$ has access to its own private dataset of size $M_i$. In Appendix C, we have characterized the smoothness $L_i$ and strong convexity parameter $\mu$ for every local function $f_i$, given its local dataset. On an Nvidia 970GTX GPU, we have used the one-versus-all technique to solve 10 independent binary classification problems (using logistic ridge regression). In the following, we focus on our two use cases, introduced in Section 4, for multiple networking scenarios.

Use Case 1: stragglers. We first assume that $N = 20$. To ensure some statistical difference among the local datasets, so as to ensure different $L_i$, we keep the samples of only 1 randomly selected class at every node. Consequently, we end up with a training task with around 300 examples in every node (a total of 5927 examples). We then consider three cost models:

- **No straggler:** $c_1 = 0.1$ and $c_i = 1$ for all $i \in [N] \setminus \{1\}$;
- **Two stragglers:** $c_1 = 0.1, c_{10} = c_{20} = 100$, and $c_i = 1$ for all $i \in [N] \setminus \{1, 10, 20\}$; and
- **Four stragglers:** $c_1 = 0.1, c_9 = c_{10} = c_{19} = c_{20} = 100$, and for all other $i \in [N], c_i = 1$.

Figure 1 illustrates the convergence of our performance measures when the sampling is optimal. SVRG-AS+ can maintain the convergence to the optimal solution for all cost models, leading to 82% cost reduction of SVRG-AS+ compared to the vanilla SVRG for the two straggler model. The significant cost reduction in Figure 1(c) is due to optimal sampling as well as better inner loop structure, as discussed in Section 3. We have reported the performance of our final solution on all digits in the Appendix.

To study the performance in nonconvex setting, we have reported in Table 1 the F1-score for the CIFAR10 dataset, trained on the VGG model. For the benchmark, we have implemented SARAH with arbitrary sampling [Horváth and Richtarík, 2019, Algorithm 3]. In all our experiments, including convex and nonconvex models and a variety of datasets, we have observed a significant gain for the network cost over the benchmarks, when we add network utility as the cost. In all cases, the convergence of SVRG-AS+ was as fast as that of SVRG. We should highlight that we did not try to optimize hyper-parameters to achieve a better F1-score in our experiments.

Table 1: F1-score of the CIFAR10 test dataset and training cost of VGG11 with cross-entropy loss, two stragglers cost model, $(\alpha_k = 0.2)_k$, $T = 15$, and 100 epochs.

<table>
<thead>
<tr>
<th>$N$</th>
<th>SVRG F1-score</th>
<th>SARAH F1-score</th>
<th>SARAH cost (x1000)</th>
<th>SVRG-AS+ F1-score</th>
<th>SVRG-AS+ cost (x1000)</th>
</tr>
</thead>
<tbody>
<tr>
<td>10</td>
<td>0.915</td>
<td>0.921</td>
<td>475.1</td>
<td>0.914</td>
<td>182.7</td>
</tr>
<tr>
<td>50</td>
<td>0.909</td>
<td>0.917</td>
<td>912.9</td>
<td>0.916</td>
<td>104.2</td>
</tr>
<tr>
<td>100</td>
<td>0.898</td>
<td>0.885</td>
<td>1657.8</td>
<td>0.882</td>
<td>79.2</td>
</tr>
</tbody>
</table>

Figure 1: Convergence results for $T = 15$, assuming digit 3 is the class 1 while all other digits are class -1. In (c) legends (0), (2), and (4) corresponds to no straggler, two straggler, and four stragglers scenarios, respectively.
We addressed the problem of minimizing the network costs associated with running a distributed optimization algorithm. In particular, we analyzed the convergence of SVRG-AS+ with arbitrary sampling and characterized the cost (in terms of the usage of network resources) of finding the solution. We then optimized the sampling probability as well as mini-batch size for SVRG-AS+ for two networking scenarios: federated learning with straggler nodes and information exchange over a shared wireless network. We have shown that our optimal design can substantially reduce the cost of running SVRG while maintaining an acceptable convergence rate. These results provide important insights to future sustainable networked artificial intelligence and machine learning over large-scale networks, such as Internet-of-Things and cyber-physical systems.
REFERENCES


