IEEE Access
Multidisciplinary | Rapid Review | Open Access Journal

Received 16 August 2023, accepted 6 September 2023, date of publication 14 September 2023,
date of current version 19 September 2023.

Digital Object Identifier 10.1109/ACCESS.2023.3315331

RESEARCH ARTICLE

Distributed Stochastic Gradient Descent With Compressed and Skipped Communication

TRAN THI PHUONG[1], LE TRIEU PHONG[2], AND KAZUHIDE FUKUSHIMA[1]
[1]KDDI Research Inc., Saitama 356-8502, Japan
[2]National Institute of Information and Communications Technology (NICT), Tokyo 184-8795, Japan

Corresponding author: Tran Thi Phuong (xph-tran@kddi.com)

The work of Le Trieu Phong was supported in part by JST CREST under Grant JPMJCR21M1; and in part by JST AIP Accelerated Program, Japan, under Grant JPMJCR22U5.

ABSTRACT This paper introduces CompSkipDSGD, a new algorithm for distributed stochastic gradient descent that aims to improve communication efficiency by compressing and selectively skipping communication. In addition to compression, CompSkipDSGD allows both workers and the server to skip communication in any iteration of the training process and reserve it for future iterations without significantly decreasing testing accuracy. Our experimental results on the large-scale ImageNet dataset demonstrate that CompSkipDSGD can save hundreds of gigabytes of communication while maintaining similar levels of accuracy compared to state-of-the-art algorithms. The experimental results are supported by a theoretical analysis that demonstrates the convergence of CompSkipDSGD under established assumptions. Overall, CompSkipDSGD could be useful for reducing communication costs in distributed deep learning and enabling the use of large-scale datasets and models in complex environments.

INDEX TERMS Compressed and skipped communication, distributed stochastic gradient descent, deep learning.

I. INTRODUCTION

Distributed stochastic gradient descent (DSGD) is a fundamental algorithm in deep learning due to its ability to handle large amounts of distributed data. This is a crucial factor in achieving superior performance in deep learning. In DSGD, multiple distributed workers compute locally on their datasets and communicate with a central parameter server to update the weight parameters of a deep neural network model. It has been established that utilizing large amounts of training data in conjunction with a large number of neural network parameters can greatly enhance learning results.

The use of DSGD and its variants is prevalent in deep learning. However, practical system-level considerations must be taken into account. One such consideration is the communication between a worker and the parameter server, which can become a bottleneck [1], [2], [3], [4], [5], [6], [7], [8], [9], [10], [11], [12].

The associate editor coordinating the review of this manuscript and approving it for publication was Yunlong Cai.

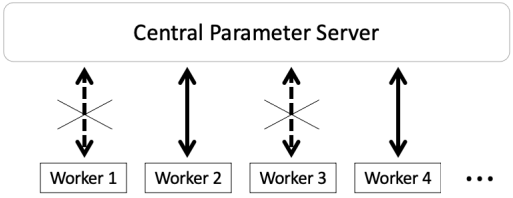

FIGURE 1. Distributed computation with unexpected skips, represented by the × marks.

To address this issue, researchers have studied techniques such as replication-based methods [13], [14], [15], [16], [17] and asynchronous optimization [1], [2], [18], [19] to deal with slow workers, also known as stragglers. It is crucial to consider these techniques and their associated trade-offs while employing DSGD in real-world applications.

The communication efficiency of DSGD has been a topic of research in the literature, with many works focusing on the compression of communication from workers [7], [11], [20], [21], [22], [23], [24], [25]. There

have been proposals to also compress communication from the parameter server [4], [26], [27], [28], [29], leading to algorithms with bidirectional compression of gradients. While unified frameworks have been proposed in [30] and [31], they do not consider momentum or server skip in the analysis and experiments. It is worth noting that momentum plays a significant role in the accurate experimentation of large scale datasets like ImageNet. Furthermore, the absence of server skip analysis is another limitation that requires further investigation.

In this context, we propose to consider a system model where, in addition to communication compression, any training worker or the central parameter server can temporarily "skip" any interaction as in Figure 1, or in other words, both workers and the server can compress the communication to the extreme. This aims at capturing unexpected events in real-world systems that prevent the machines from communicating with each other. In this setting, such iteration skips allow a worker to decide whether to participate in the training process based on its current connectivity, and also allow the central parameter server to temporarily ignore slow workers in any training iteration.

It is important to note that having such communication skips to improve communication efficiency should not come at the cost of poor learning performance. Therefore, it is crucial to have a good learning performance in terms of testing accuracies, backed up by rigorous theoretical analysis.

A. OUR CONTRIBUTION

We present and evaluate CompSkipDSGD as a derived form of DSGD handling both communication skip and communication efficiency. Details are as follows.

- Both the parameter server and the workers in CompSkipDSGD can skip any step of the training process. In addition, the workers and the server can compress their communication. Concretely compared with the state-of-the-art on the same level of ImageNet top-1 accuracy, CompSkipDSGD saves at least 328 gigabytes of communication. These properties make CompSkipDSGD advantageous from the viewpoint of network latency.
- The (top-1) accuracies on the large-scale ImageNet dataset of CompSkipDSGD are at least 76%, and hence comparable with baseline results, as shown by various experiments in Section III. These experimental results are theoretically backed up by a mathematical proof of convergence for CompSkipDSGD presented in Section II.

As seen in Table 1, the primary contribution of our algorithm is its pioneering approach of incorporating skips for both workers and servers, which represents a novel aspect that has not been explored in prior literature. This innovative feature sets our algorithm apart from existing methods and introduces a new dimension to the field. The experiments conducted on the ImageNet dataset demonstrate the effectiveness of our algorithm in handling such large-scale datasets.

TABLE 1. CompSkipDSGD and its predecessors.

Paper	Server/Workers Skip Possible?	ImageNet Top-1 Accuracy
Original distributed SGD	no	76.27%
[26]	no	72.77%
[4]	no	76.77%
[27]	no	76.09%
[23], [30], [34]	no	n/a
[35]	no	74.34%
Our CompSkipDSGD	yes	76.40%

The description of CompSkipDSGD is given in Algorithm 1 with explanations in Section II. Technically, CompSkipDSGD, while basing on [4], [32], [33], adds a boolean flag called skip to the workers and the server to indicate whether the participants skip a training iteration or not. The flag can be determined by probabilistic events, and in the experiments by random coin tosses. This flag makes the theoretical analyses for previous works such as in [4], [32], and [33] not applicable to CompSkipDSGD anymore because the compression rate of communication becomes 0 when skip is set to true. In addition, when a participant skips, the error-feedback technique as previously used in [4], [32], and [33] turns out to be potentially problematic, because full errors from past training iterations can become dominated in the current and future ones. We can resolve this issue by using a little trick that involves utilizing a constant hyper-parameter γ (which can be set to 0.9 by default). This hyper-parameter is multiplied to the errors so that past errors can be properly scaled down as the training process continues.

Our CompSkipDSGD not only grants the flexibility for a worker or the server to skip tasks but also ensures synchronization with other workers. This fundamental characteristic distinguishes CompSkipDSGD from asynchronous distributed algorithms. Studies such as [36], [37] have shown that synchronous algorithms generally achieve higher levels of accuracy when compared to their asynchronous counterparts. In line with this body of research, we adopt a synchronous approach in this paper.

Our CompSkipDSGD exhibits resemblances to distributed algorithms employed in time-varying graph scenarios [38], [39]. Nevertheless, it is crucial to highlight that there exist distinct differences and advantages associated with our approach compared to traditional distributed algorithms on time-varying graphs. One notable distinction is the inclusion of a central server in our method, whereas distributed algorithms on time-varying graphs typically operate without a central server. The presence of a central server simplifies the synchronization process during training, contributing to the convenience and efficiency of our approach.

II. CompSkipDSGD AND ITS ANALYSIS

The description of CompSkipDSGD is in Algorithm 1 in which there are M workers connected to a central parameter server. In a worker, line 5 computes the stochastic gradient

Algorithm 1 Distributed SGD With Compressed and Skipped Communication (CompSkipDSGD)

1: **Parameters:** $\eta_t, c_t, c'_t, \tilde{c}_t, \beta_{t,i}, \tilde{\beta}_t, \mu, \gamma$
2: **Initialize:** $x_0 \in \mathbb{R}^d$; $m_{-1,i} = e_{0,i} = 0 \in \mathbb{R}^d$; $\tilde{e}_0 = 0 \in \mathbb{R}^d$
3: **for** t **from** 0 **to** $T-1$ **do**
4: • **The i-th worker** ($1 \le i \le M$)**:**
5: $g_{t,i} = \nabla \ell(x_t, \xi_{t,i})$ for data $\xi_{t,i}$
6: $m_{t,i} = \mu \cdot m_{t-1,i} + g_{t,i}$
7: $p_{t,i} = \mu \cdot m_{t,i} + g_{t,i} + \gamma \cdot e_{t,i}$
8: ∘ if $\mathsf{skip}_{t,i}$ is false: push $\Delta_{t,i} = \mathsf{sign}_{\beta_{t,i}}(p_{t,i}) \in \mathbb{R}^d$ to server
9: ∘ if $\mathsf{skip}_{t,i}$ is true: set $\Delta_{t,i} = 0 \in \mathbb{R}^d$, and send skip to server
10: receive $\tilde{\Delta}_t$ from server
11: $x_{t+1} = x_t - \eta_t \cdot \tilde{\Delta}_t$
12: $e_{t+1,i} = p_{t,i} - c_t \cdot \Delta_{t,i}$
13: • **Central parameter server:**
14: obtain $\Delta_{t,i}$ that is transmitted by workers, namely $i \in S_t = \{i : \mathsf{skip}_{t,i} \text{ is false}\}$
15: compute $\tilde{p}_t = \gamma \cdot \tilde{e}_t + \frac{1}{|S_t|} \sum_{i \in S_t} c'_t \cdot \Delta_{t,i}$
16: ∘ if $\mathsf{serverSkip}_t$ is false: push $\tilde{\Delta}_t = \mathsf{sign}_{\tilde{\beta}_t}(\tilde{p}_t)$ to workers
17: ∘ if $\mathsf{serverSkip}_t$ is true: set $\tilde{\Delta}_t = 0 \in \mathbb{R}^d$ and send skip to workers
18: $\tilde{e}_{t+1} = \tilde{p}_t - \tilde{c}_t \cdot \tilde{\Delta}_t$
19: **end for**

of a data batch. Line 6 adds stochastic momentum to the gradient. Line 7 applies Nesterov momentum added with the past error scaled by the hyper-parameter γ. The function of the form $\mathsf{sign}_\beta(p)$ at line 8 (and line 16) transforms p into a vector of signs in which each component is kept the same with probability β, and otherwise zero with probability $1-\beta$. Concretely, if $p = (p_1, \ldots, p_d) \in \mathbb{R}^d$ then $\mathsf{sign}_\beta(p) = (sp_1, \ldots, sp_d) \in \{-1, 0, 1\}^d$ in which

$$sp_i = \begin{cases} \mathsf{sign}(p_i) & \text{with probability } \beta \\ 0 & \text{with probability } 1-\beta \end{cases}$$

In lines 8 and 9, there is a boolean flag $\mathsf{skip}_{t,i}$ of worker i in iteration t: if $\mathsf{skip}_{t,i}$ is false then the gradient signs are sent to the server; and if $\mathsf{skip}_{t,i}$ is true then essentially nothing is sent to the server. Therefore, the flag $\mathsf{skip}_{t,i}$ can be used by worker i to control the communication with the server at each iteration. Line 12 updates the local error at each worker, with the error-learning rate c_t. When $\mathsf{skip}_{t,i}$ is true, in line 12 we have $e_{t+1,i} = p_{t,i}$ which means that the error fed to iteration $t+1$ contains the full gradient of iteration t, scaled by a factor γ controlling the effect of such error, particularly in future iterations as discussed above.

Line 15 averages all $\Delta_{t,i}$, and adds possible server error. At lines 16 and 17, the boolean flag $\mathsf{serverSkip}_t$ determines whether the server sends back the accumulated and processed gradient signs to all of the workers. If $\mathsf{serverSkip}_t$ is true, then $\tilde{e}_{t+1} = \tilde{p}_t$ which as above means that the full gradient is reused for the next iteration. Line 18 updates the server error using a hyper-parameter $\tilde{c}_t$. Later, we take $c_t = \tilde{c}_t = c$ for a small constant c.

One approach to reducing communication between the workers and the parameter server is to have them set the boolean flags $\mathsf{skip}_{t,i}$ and $\mathsf{serverSkip}_t$ uniformly at random. By doing so, they can independently determine whether or not to communicate based on a random selection process. This randomness helps distribute the communication load more evenly among the workers and alleviates potential bottlenecks.

In addition to random flag setting, another technique to optimize communication is by considering local indicators such as communication bandwidth. Each worker and the parameter server can monitor their own communication capabilities, such as available bandwidth or network congestion, and use this information to make informed decisions about participating in the training process.

By taking into account local indicators, the workers and the parameter server can dynamically adjust their probability of participating in communication. For instance, if a worker or the parameter server detects a high communication bandwidth and low congestion, they might increase their probability of engaging in communication, as the network conditions are favorable. Conversely, if they observe limited bandwidth or high congestion, they could decrease their probability of participation to avoid exacerbating communication issues.

This adaptive approach allows the workers and the parameter server to make efficient use of their resources while balancing the training workload. By leveraging randomness and considering local indicators, they can collectively reduce unnecessary communication and improve the overall efficiency of the training process.

The algorithm CompSkipDSGD can encompass the following algorithms in [26] and [27] in the literature as special cases:

- It covers signSGD with majority vote in [26] as a special case by setting $\mu = \gamma = 0$, $\mathsf{skip}_{t,i} = \mathsf{serverSkip}_t =$ false for all t, i.
- It covers the dropSignSGD in [27] by setting $\gamma = 1$, and $\mathsf{skip}_{t,i} = \mathsf{serverSkip}_t =$ false for all t, i.

Having the boolean flags for skipping the communication as in CompSkipDSGD adds more flexibility to the workers and the server, while simultaneously reducing the communication costs as shown in the experiments.

Let us proceed with the theoretical analysis of the algorithm, ensuring that it converges mathematically. Let $\|\cdot\|$ be the Euclidean norm, and $\langle\cdot,\cdot\rangle$ be the inner product. Given a loss function $\ell : \mathbb{R}^d \times \Xi \to \mathbb{R}^+$ which can be non-convex, set $f(x) = \mathbf{E}_\xi[\ell(x,\xi)]$ where $x \in \mathbb{R}^d$ represents the parameters of a neural network, and ξ is a data batch. We use below assumptions [4], [5], [27], [33] for the theoretical analysis of CompSkipDSGD.

Assumption 1: The value $f^\star = \inf_{x\in\mathbb{R}^d} f(x) < \infty$ exists. The function f is differentiable, and for a positive number L:

$$\|\nabla f(x) - \nabla f(y)\| \le L\|x-y\|, \quad \forall x, y \in \mathbb{R}^d, \tag{1}$$

$$f(x) \le f(y) + \langle \nabla f(y), x-y\rangle + \frac{L}{2}\|x-y\|^2. \tag{2}$$

Assumption 2: $\mathbf{E}_t[g_{t,i}] = \nabla f(x_t)$ and $\exists\sigma$, $\mathbf{E}_t[\|g_{t,i} - \nabla f(x_t)\|^2] \le \sigma^2$ where $\mathbf{E}_t$ is the expectation at step t.

Assumption 3: $\|\nabla f(x_t)\|^2 \le \omega^2$ for a constant ω.

Assumption 4: $\{g_{t,i} - \nabla f(x_t)\}_{1\le i\le M}$ are independently random vectors.

The theoretical convergence of CompSkipDSGD is ensured by the following result.

Theorem 1: Given Assumptions 1-4, and $c_t = \tilde{c}_t = c\ \forall t \ge 0$ for some $c > 0$, there is a sequence $\{\eta_t\}$ such that

$$\min_{0\le t\le T-1}\mathbf{E}[\|\nabla f(x_t)\|^2] \le O\left(\frac{1}{\sqrt{T}}\right),$$

namely the gradient components of f approaches 0 when the number of steps T becomes large.

Proof: Given $x_t, \tilde{e}_t, \eta_t, e_{t,i}$ from Algorithm 1, let $\eta_{-1} = 0$ and for $t \ge 0$ we define the the sequence $\{\tilde{x}_t\}$ as follows

$$\tilde{x}_t = x_t - \frac{\eta_{t-1}}{c}\left(\tilde{e}_t + \frac{1}{M}\sum_{i=1}^M e_{t,i}\right).$$

Then by Lemma 1 (see the appendix), we have

$$\tilde{x}_{t+1} = \tilde{x}_t - \frac{\eta_t}{cM}\sum_{i=1}^M g_{t,i}.$$

Using (2), we have,

$$\begin{aligned}
&\mathbf{E}_t[f(\tilde{x}_{t+1})] \le f(\tilde{x}_t) \\
&\quad + \langle \nabla f(\tilde{x}_t), \mathbf{E}_t[\tilde{x}_{t+1} - \tilde{x}_t]\rangle + \frac{L}{2}\mathbf{E}_t[\|\tilde{x}_{t+1} - \tilde{x}_t\|^2] \\
&= f(\tilde{x}_t) - \frac{\eta_t}{c}\left\langle \nabla f(\tilde{x}_t), \mathbf{E}_t\left[\frac{1}{M}\sum_{i=1}^M g_{t,i}\right]\right\rangle \\
&\quad + \frac{L\eta_t^2}{2c^2}\mathbf{E}_t\left[\left\|\frac{1}{M}\sum_{i=1}^M g_{t,i}\right\|^2\right] \qquad (3) \\
&= f(\tilde{x}_t) - \frac{\eta_t}{c}\langle \nabla f(\tilde{x}_t), \nabla f(x_t)\rangle \\
&\quad + \frac{L\eta_t^2}{2c^2}\mathbf{E}_t\left[\left\|\frac{1}{M}\sum_{i=1}^M g_{t,i}\right\|^2\right], \qquad (4)
\end{aligned}$$

where (3) and (4) are by Lemma 1 and Assumption 2 respectively. Additionally, given the expectation on the stochastic gradient $\mathbf{E}_t[g_{t,i}] = \nabla f(x_t)$ by Assumption 2 which yields $\mathbf{E}_t[\frac{1}{M}\sum_{i=1}^M g_{t,i}] - \nabla f(x_t) = 0$, we have

$$\begin{aligned}
\mathbf{E}_t\left[\left\|\frac{1}{M}\sum_{i=1}^M g_{t,i}\right\|^2\right] &= \mathbf{E}_t\left[\|\nabla f(x_t)\|^2\right] \\
&\quad + 2\mathbf{E}_t\Bigg\langle \underbrace{\frac{1}{M}\sum_{i=1}^M g_{t,i} - \nabla f(x_t)}_{=0 \text{ after applying } \mathbf{E}_t}, \nabla f(x_t)\Bigg\rangle \\
&\quad + \mathbf{E}_t\left[\left\|\frac{1}{M}\sum_{i=1}^M g_{t,i} - \nabla f(x_t)\right\|^2\right], \\
&= \mathbf{E}_t\left[\|\nabla f(x_t)\|^2\right] \\
&\quad + \mathbf{E}_t\left[\left\|\frac{1}{M}\sum_{i=1}^M g_{t,i} - \nabla f(x_t)\right\|^2\right],
\end{aligned}$$

combining with (4), we have

$$\begin{aligned}
\mathbf{E}_t[f(\tilde{x}_{t+1})] &\le f(\tilde{x}_t) - \frac{\eta_t}{c}\langle \nabla f(\tilde{x}_t), \nabla f(x_t)\rangle \\
&\quad + \frac{L\eta_t^2}{2c^2}\|\nabla f(x_t)\|^2 \\
&\quad + \frac{L\eta_t^2}{2c^2}\mathbf{E}_t\left[\left\|\frac{1}{M}\sum_{i=1}^M g_{t,i} - \nabla f(x_t)\right\|^2\right].
\end{aligned}$$

Using Assumption 4,

$$\begin{aligned}
\mathbf{E}_t\left[\left\|\sum_{i=1}^M (g_{t,i} - \nabla f(x_t))\right\|^2\right] &= \sum_{i=1}^M \mathbf{E}_t\left[\|g_{t,i} - \nabla f(x_t)\|^2\right] \\
&\le M\sigma^2.
\end{aligned}$$

Therefore

$$\mathbf{E}_t\left[\left\|\frac{1}{M}\sum_{i=1}^M (g_{t,i} - \nabla f(x_t))\right\|^2\right] \le \frac{\sigma^2}{M}. \tag{5}$$

By (5), we obtain

$$\begin{aligned}
\mathbf{E}_t[f(\tilde{x}_{t+1})] &\le f(\tilde{x}_t) - \frac{\eta_t}{c}\langle \nabla f(\tilde{x}_t), \nabla f(x_t)\rangle \\
&\quad + \frac{L\eta_t^2}{2c^2}\|\nabla f(x_t)\|^2 + \frac{L\eta_t^2\sigma^2}{2c^2M}. \qquad (6)
\end{aligned}$$

In addition, we have

$$\begin{aligned}
&-\langle \nabla f(\tilde{x}_t), \nabla f(x_t)\rangle \\
&= \langle \nabla f(x_t) - \nabla f(\tilde{x}_t), \nabla f(x_t)\rangle - \langle \nabla f(x_t), \nabla f(x_t)\rangle \\
&= \langle \nabla f(x_t) - \nabla f(\tilde{x}_t), \nabla f(x_t)\rangle - \|\nabla f(x_t)\|^2 \\
&\le \frac{1}{2}\|\nabla f(x_t)\|^2 + \frac{1}{2}\|\nabla f(x_t) - \nabla f(\tilde{x}_t)\|^2 - \|\nabla f(x_t)\|^2 \\
&= -\frac{1}{2}\|\nabla f(x_t)\|^2 + \frac{1}{2}\|\nabla f(x_t) - \nabla f(\tilde{x}_t)\|^2 \\
&\le -\frac{1}{2}\|\nabla f(x_t)\|^2 + \frac{L^2}{2}\|x_t - \tilde{x}_t\|^2, \qquad (7)
\end{aligned}$$

where the last inequality is by (1). By the setting of the sequence $\{\tilde{x}_t\}$, we obtain

$$\begin{aligned}
\|x_t - \tilde{x}_t\|^2 &= \frac{\eta_{t-1}^2}{c^2}\left\|\tilde{e}_t + \frac{1}{M}\sum_{i=1}^{M} e_{t,i}\right\|^2 \\
&\le \frac{\eta_{t-1}^2}{c^2} G^2 U, \qquad (8)
\end{aligned}$$

where the last inequality is by Lemma 2. Given (8), the inequality (7) provides us with

$$-\langle \nabla f(\tilde{x}_t), \nabla f(x_t)\rangle \le -\frac{1}{2}\|\nabla f(x_t)\|^2 + \frac{\eta_{t-1}^2}{c^2} G^2 L^2\, U.$$

Therefore, by (6), we have

$$\begin{aligned}
\mathbf{E}_t[f(\tilde{x}_{t+1})] \le f(\tilde{x}_t) &- \left(\frac{\eta_t}{2c} - \frac{L\eta_t^2}{2c^2}\right)\|\nabla f(x_t)\|^2 \\
&+ \frac{L\eta_t^2\sigma^2}{2c^2 M} + \frac{\eta_t \eta_{t-1}^2 G^2 L^2 U}{c^3}
\end{aligned}$$

which yields the following

$$\begin{aligned}
&\left(\frac{\eta_t}{2c} - \frac{L\eta_t^2}{2c^2}\right)\|\nabla f(x_t)\|^2 \\
&\quad \le \mathbf{E}[f(\tilde{x}_t) - f(\tilde{x}_{t+1})] \\
&\qquad + \frac{L\eta_t^2\sigma^2}{2c^2 M} + \frac{\eta_t \eta_{t-1}^2 G^2 L^2 U}{c^3}.
\end{aligned}$$

Let $\eta_t \le \frac{c}{2L}$, and $\eta^* = \min\{\eta_t\}_t$. Then

$$\begin{aligned}
\frac{\eta^*}{4c}\|\nabla f(x_t)\|^2 &\le \frac{\eta_t}{4c}\|\nabla f(x_t)\|^2 \\
&\le \mathbf{E}[f(\tilde{x}_t) - f(\tilde{x}_{t+1})] \\
&\quad + \frac{L\eta_t^2\sigma^2}{2c^2 M} + \frac{\eta_t \eta_{t-1}^2 G^2 L^2 U}{c^3}.
\end{aligned}$$

Therefore

$$\begin{aligned}
\|\nabla f(x_t)\|^2 \le \frac{4c}{\eta^*}&\mathbf{E}[f(\tilde{x}_t) - f(\tilde{x}_{t+1})] \\
&+ \frac{2L\eta_t^2\sigma^2}{\eta^* cM} + \frac{4\eta_t \eta_{t-1}^2 G^2 L^2 U}{\eta^* c^2}.
\end{aligned}$$

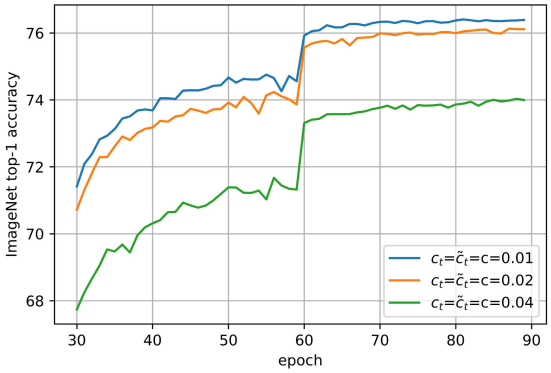

FIGURE 2. Effect of changing hyper-parameters c_t and $\tilde{c}_t$, with fixed skip probabilities of 0.3 (for all workers) and 0.1 (for server).

Using the following

$$\begin{aligned}
\sum_{t=0}^{T-1}\frac{4c}{\eta^*}\mathbf{E}[f(\tilde{x}_t) - f(\tilde{x}_{t+1})] &= \frac{4c}{\eta^*}\left(f(\tilde{x}_0) - f(\tilde{x}_T)\right) \\
&\le \frac{4c}{\eta^*}\left(f(\tilde{x}_0) - f^\star\right),
\end{aligned}$$

we obtain

$$\begin{aligned}
\sum_{t=0}^{T-1}\mathbf{E}[\|\nabla f(x_t)\|^2] \le \frac{4c}{\eta^*}\left(f(\tilde{x}_0) - f^\star\right) &+ \sum_{t=0}^{T-1}\frac{2L\eta_t^2\sigma^2}{\eta^* cM} \\
&+ \sum_{t=0}^{T-1}\frac{4\eta_t\eta_{t-1}^2 G^2 L^2 U}{\eta^* c^2}.
\end{aligned}$$

If $\eta_0 = 1/\sqrt{T}$ and $\eta_t = \eta_0\gamma^{-t}$, we have $\sum_{t=0}^{T-1}\eta_t = \eta_0(1-\gamma^{-T})/(1-\gamma^{-1})$ and $\sum_{t=0}^{T-1}\eta_t^2 = \eta_0^2(1-\gamma^{-2T})/(1-\gamma^{-2})$. Therefore, $\frac{1}{T}\sum_{t=0}^{T-1}\eta_t = O(1/\sqrt{T})$ and $\frac{1}{T}\sum_{t=0}^{T-1}\eta_t^2 = O(1/T)$ provided that $\gamma \lesssim 1$ such as $\gamma = 1 - \frac{1}{T+1}$. In addition, $1/(T\eta^*) = 1/(T\eta_0) = 1/\sqrt{T}$ and $\eta_t/\eta^* \approx 1$ by such choice of γ. These conditions give us

$$\begin{aligned}
&\frac{1}{T}\sum_{t=0}^{T-1}\mathbf{E}[\|\nabla f(x_t)\|^2] \\
&\quad \le O\left(\frac{4c\,(f(\tilde{x}_0) - f^\star)}{\sqrt{T}} + \frac{2L\sigma^2}{cM\sqrt{T}} + \frac{4G^2L^2U}{c^2T}\right).
\end{aligned}$$

The number of clients M, those skipped at an iteration, and the compression parameters $(\beta_{t,i}, \tilde{\beta}_t)$ only affect the higher term $1/T$ in the inequality. Therefore, the convergence rate is determined by the lower terms of $O\left(\frac{1}{\sqrt{T}}\right)$, as claimed in the theorem statement. ■

III. EXPERIMENTS

We conduct experiments with the ImageNet dataset [40] using ResNet-50 [41] which has the number of parameters $d = 25,557,032$. The PyTorch codes associated with [26] and [42] are modified with necessary changes for Comp-SkipDSGD. As in previous works, we select the number of distributed workers $M = 7$, the batch size 128, and

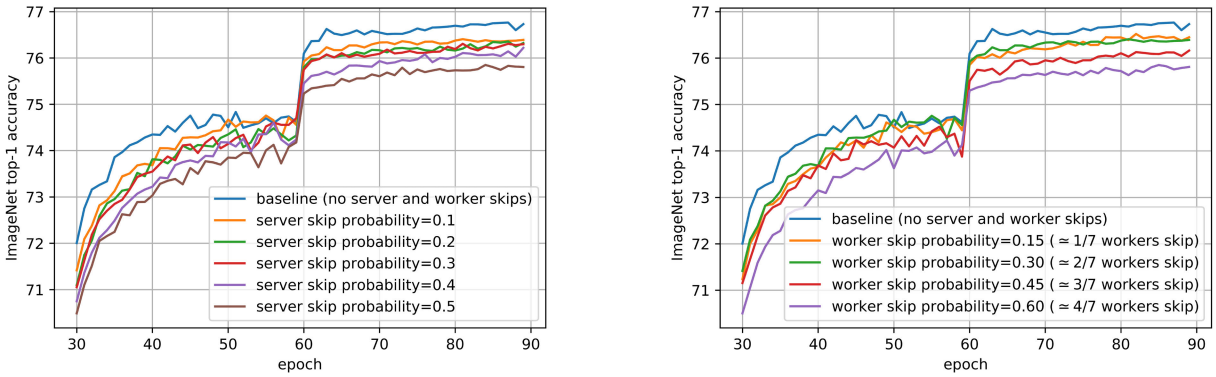

FIGURE 3. Effects of hyper-parameters, and communication skips (of the server and workers) on learning accuracies. (a) Effect of various server skip probabilities, given a fixed worker skip probability of 0.3. (b) Effect of various worker skip probabilities, given a fixed server probability of 0.1.

momentum $\mu = 0.9$. We set $\gamma = 0.9$, $\beta_{t,i} = 0.3$, and $\tilde{\beta}_t = 0.5$ in the experiments if not otherwise stated.

A. THE EFFECT OF HYPER-PARAMETERS C_T AND $\tilde{C}_T$

In the experiments, we choose $c_t = \tilde{c}_t = c$, in which $c \in \{10^{-2}, 2 \cdot 10^{-2}, 4 \cdot 10^{-2}\}$, and fixed skip probabilities of 0.3 (for all workers) and 0.1 (for server). As seen in Figure 2, the choice of $c = 10^{-2}$ gives a better graph, with final accuracy of 76.40% instead of 76.12% (when $c = 2 \cdot 10^{-2}$) and 74.03% (when $c = 4 \cdot 10^{-2}$). Therefore, for other experiments, we simply stick with this choice of $c = 10^{-2}$.

B. THE EFFECT OF SERVER SKIP

In Figure 3(a), we vary the probabilities that the server skips, while maintaining a constant probability that a worker skips, to examine the effect of server skip on the testing accuracy. Specifically, the probability that the server skips is set to 0.1, 0.2, 0.3, 0.4, and 0.5; while a worker skips with a probability of 0.3. As expected, the testing accuracy decreases as the server skip probability increases as shown in Figure 3(a). The gained accuracies are 76.40%, 76.35%, 76.30%, 76.21%, and 75.84% respectively. Therefore, if the server skips with probability ≤ 0.4, the final accuracy can still be at least 76%.

C. THE EFFECT OF WORKER SKIP

In Figure 3(b), we vary the probabilities that the workers skip while maintaining a constant probability that the server skips, to examine the effect of worker skip on the testing accuracy. Specifically, the probability that the server skips is set to 0.1; while a worker skips with probability set in $\{0.15, 0.30, 0.45, 0.60\}$. Because we consider 7 workers, this setting means that in each iteration, there are $7 \times \{0.15, 0.30, 0.45, 0.60\} = \{1.05, 2.10, 3.15, 4.20\}$ workers who do skip on the average. In other words, there are expectedly more than $\{1, 2, 3, 4\}$ skip workers in each iteration. Figure 3(b) depicts the accuracy graphs corresponding to these settings, showing that the accuracy decreases (as expected) if more workers skip. Nonetheless, the accuracy decline is not quite aggressive, as in the worst case of 60% skip workers in each iteration, the final accuracy is still 75.85%, compared with 76.76% of the baseline with no skips at all. In addition, in other cases, the final accuracies are at least 76%.

Although the algorithm converges to a local minimum, there is a decline in testing accuracy, suggesting that the resulting model exhibits poorer generalization capabilities. This decline in testing accuracy can be attributed to the reduced amount of data utilized during training due to the skipping mechanisms.

TABLE 2. Saved communication compared to the algorithm in [27] on approximately identical level of testing accuracy.

(Worker, Server) Skip Probability	Saved Communication	Testing Accuracy
(0.3, 0.1)	328 GB	76.40%
(0.3, 0.2)	410 GB	76.35%
(0.3, 0.3)	492 GB	76.30%
(0.3, 0.4)	574 GB	76.21%

D. COMPARISON ON COMMUNICATION COSTS

Besides communication skips (which are not allowed by previous works such as original DSGD and [4], [7], [20], [26], [27]), CompSkipDSGD is relatively communication-efficient. We set the skip probability of workers to 0.3, and the skip probability of the server to 0.1 respectively. Therefore, the probabilities of non-skip communication in each iteration become 0.7 (for workers), and 0.9 (for the server, respectively). Combined with the fact that each worker only sends $0.3d$ bits of gradients in our setting, the number of bits each worker needs to transmit is $0.7 \times 0.3d = 0.21d$ on average. Similar computations hold for the server, yielding at most $0.27Md$ bits, in which M is the number of workers.

Let us compute a concrete amount of communication saved by CompSkipDSGD. Admitting a negligible decline (0.09%) of accuracy from 76.09% in [27] to 76.00% in CompSkipDSGD, the server saves $0.3Md - 0.27Md =$

$0.03Md$ bits in each iteration. With the number of workers $M = 7$ and the number of epochs of 90 to reach such accuracies on the ImageNet dataset, the number of iterations becomes 90 (epochs) $\times 1359 = 122,310$ iterations because the number of iterations in one epoch is 1359. Therefore, the number of bits saved is $122,310 \times 0.03Md = 3669.3Md = 3669.3 \times 7 \times 25,557,032 = 6.5643 \times 10^{11}$ (bits), which is converted into 82 gigabytes approximately. Similarly, the communication amount saved by CompSkipDSGD for M workers is of the form $122,310 \times (0.3 - 0.21)d \times M = 1.9693 \times 10^{12}$ (bits), which is converted into 246 gigabytes approximately. Some more numbers on the saved communication cost are given in Table 2. Summing up, our CompSkipDSGD saves hundreds of gigabytes when compared with the algorithm in [27] on almost the same level of accuracy.

IV. CONCLUSION

We propose a new algorithm, called CompSkipDSGD, which enables both workers and the server to temporarily skip communication while maintaining comparable accuracy on a large-scale benchmark dataset, resulting in better communication efficiency. We demonstrate that CompSkipDSGD mathematically converges. Our approach of incorporating communication skips with an internal state to improve accuracy could potentially impact the development of other algorithms in the future.

In addition, it is important to acknowledge that real-life optimization problems encountered in practical environments often demand a multi-faceted approach. Such challenges typically necessitate the utilization of various existing algorithms, and CompSkipDSGD can be regarded as a valuable addition to the repertoire of available techniques.

APPENDIX HELPING LEMMAS

Below are helping lemmas for the proof of Theorem 1.

Lemma 1: Let $c_t = \tilde{c}_t = c > 0$, $\eta_t = \gamma^{-1}\eta_{t-1}$, $c'_t/|S_t| = c/M$, and $\tilde{x}_t = x_t - \frac{\eta_{t-1}}{c}\left(\tilde{e}_t + \frac{1}{M}\sum_{i=1}^M e_{t,i}\right)$. Then

$$\tilde{x}_{t+1} = \tilde{x}_t - \frac{\eta_t}{cM}\sum_{i=1}^M g_{t,i}.$$

Proof: By definition

$$\begin{aligned}
\tilde{x}_{t+1} &= x_{t+1} - \frac{\eta_t}{c}\left(\tilde{e}_{t+1} + \frac{1}{M}\sum_{i=1}^M e_{t+1,i}\right) \\
&= x_t - \frac{\eta_t}{c}\Big(\frac{c}{M}\sum_{i\in S_t}\mathsf{sign}_{\beta_{t,i}}(p_{t,i}) + \gamma\cdot\tilde{e}_t\Big) \\
&\quad - \frac{\eta_t}{cM}\sum_{i=1}^M e_{t+1,i} \\
&= x_t - \frac{\eta_t}{cM}\sum_{i\in S_t}(c\cdot\mathsf{sign}_{\beta_{t,i}}(p_{t,i}) + e_{t+1,i}) \\
&\quad - \frac{\gamma\cdot\eta_t}{c}\tilde{e}_t - \frac{\eta_t}{cM}\sum_{i\notin S_t} e_{t+1,i} \\
&= x_t - \frac{\eta_t}{cM}\sum_{i\in S_t} p_{t,i} - \frac{\gamma\cdot\eta_t}{c}\tilde{e}_t - \frac{\eta_t}{cM}\sum_{i\notin S_t} p_{t,i} \\
&= x_t - \frac{\gamma\cdot\eta_t}{c}\tilde{e}_t - \frac{\eta_t}{cM}\sum_{i=1}^M (g_{t,i} + \gamma e_{t,i}) \\
&= x_t - \frac{\gamma\cdot\eta_t}{c}\left(\tilde{e}_t + \frac{1}{M}\sum_{i=1}^M e_{t,i}\right) - \frac{\eta_t}{cM}\sum_{i=1}^M g_{t,i} \\
&= x_t - \frac{\eta_{t-1}}{c}\left(\tilde{e}_t + \frac{1}{M}\sum_{i=1}^M e_{t,i}\right) - \frac{\eta_t}{cM}\sum_{i=1}^M g_{t,i} \\
&= \tilde{x}_t - \frac{\eta_t}{cM}\sum_{i=1}^M g_{t,i}
\end{aligned}$$

which finishes the proof. ∎

For the purpose of simplifying the theoretical analysis of the algorithm, we specifically set the momentum parameter to 0 in the proof. This enables us to focus on the core principles and mathematical reasoning underlying the algorithm, without the added complexity introduced by non-zero momentum values. It is important to note that while we choose a specific value in the proof for theoretical convenience, the experimentation phase allows for a broader exploration of different momentum parameter settings.

Lemma 2: Let $G^2 = \sigma^2 + \omega^2$ for constants σ and ω given in Assumptions 2 and 3. Additionally, assume the momentum parameter $\mu = 0$. There is a bound U such that

$$\left\|\tilde{e}_t + \frac{1}{M}\sum_{i=1}^M e_{t,i}\right\|^2 \le G^2\, U.$$

Proof: We have

$$\left\|\tilde{e}_{t+1} + \frac{1}{M}\sum_{i=1}^M e_{t+1,i}\right\|^2 \le 2\|\tilde{e}_{t+1}\|^2 + \frac{2}{M}\sum_{i=1}^M\|e_{t+1,i}\|^2. \tag{9}$$

Additionally,

$$\begin{aligned}
\frac{1}{M}\sum_{i=1}^M\|e_{t+1,i}\|^2 &= \frac{1}{M}\sum_{i\in S_t}\|c_t\mathsf{sign}_{\beta_{t,i}}(p_{t,i}) - p_{t,i}\|^2 \\
&\quad + \frac{1}{M}\sum_{i\notin S_t}\|p_{t,i}\|^2 \qquad (10) \\
&\le \frac{1}{M}\sum_{i\in S_t}(1-\delta_{\beta_{t,i}})\|p_{t,i}\|^2 + \frac{1}{M}\sum_{i\notin S_t}\|p_{t,i}\|^2 \\
&\qquad (11) \\
&\le \frac{(1-\delta)}{M}\sum_{i\in S_t}\|p_{t,i}\|^2 + \frac{1}{M}\sum_{i\notin S_t}\|p_{t,i}\|^2. \qquad (12)
\end{aligned}$$

where equality (10) is by the setting of $e_{t+1,i}$ and $p_{t,i}$ in Algorithm 1; (11) and $0 < \delta_{\beta_{t,i}} < 1$ are owing to Lemma 1 of [27]; and (12) is by setting $\delta \le \min\{\delta_{\beta_{t,i}}\}$. Continuing with

the above inequality,

$$\begin{aligned}(12)\leq&\frac{(1-\delta)}{M}\sum_{i\in S_t}\|g_{t,i}+\gamma\cdot e_{t,i}\|^2\\&+\frac{1}{M}\sum_{i\notin S_t}\|g_{t,i}+\gamma\cdot e_{t,i}\|^2 \qquad (13)\\\leq&\frac{(1-\delta)(1+\lambda)\gamma^2}{M}\sum_{i\in S_t}\|e_{t,i}\|^2\\&+\frac{(1-\delta)(1+1/\lambda)}{M}\sum_{i\in S_t}\|g_{t,i}\|^2\\&+\frac{(1+\lambda)\gamma^2}{M}\sum_{i\notin S_t}\|e_{t,i}\|^2\\&+\frac{(1+1/\lambda)}{M}\sum_{i\notin S_t}\|g_{t,i}\|^2 \qquad (14)\end{aligned}$$

for all $\lambda\geq 0$, where (14) is by Young inequality with $\lambda>0$. In addition,

$$\begin{aligned}(14)\leq&\frac{(1-\delta)(1+\lambda)}{M}\sum_{i\in S_t}\|e_{t,i}\|^2\\&+\frac{(1-\delta)(1+1/\lambda)}{M}\sum_{i\in S_t}\|g_{t,i}\|^2\\&+\frac{(1-\delta_\gamma)(1+\lambda)}{M}\sum_{i\notin S_t}\|e_{t,i}\|^2\\&+\frac{(1+1/\lambda)}{M}\sum_{i\notin S_t}\|g_{t,i}\|^2 \qquad (15)\\\leq&(1-\delta)(1+\lambda)\left(\frac{1}{M}\sum_{i=1}^{M}\|e_{t,i}\|^2\right)\\&+(1+1/\lambda)G^2, \qquad (16)\end{aligned}$$

where (15) is by $\gamma\leq 1$ and $\delta_\gamma=1-\gamma^2$; and (16) is by $1-\delta\leq 1$ and by setting $\delta\leq\delta_\gamma$. Note that inequality (16) is of the form

$$a_{t+1}\leq\alpha a_t+\beta, \qquad (17)$$

where $a_{t+1}=\frac{1}{M}\sum_{i=1}^{M}\|e_{t+1,i}\|^2$, and

$$\alpha=(1-\delta)(1+\lambda),\ \beta=(1+1/\lambda)G^2. \qquad (18)$$

Applying Lemma 1 of [32], we obtain

$$a_{t+1}\leq\beta\sum_{j=0}^{t}\alpha^j, \qquad (19)$$

By choosing $\lambda=\frac{\delta}{2(1-\delta)}$, we get

$$\beta=\frac{(2-\delta)G^2}{\delta},\ \alpha=1-\frac{\delta}{2}. \qquad (20)$$

Because $0<\alpha<1$, we obtain $\sum_{j=0}^{t}\alpha^j\leq\sum_{j\geq 0}\alpha^j=\frac{1}{1-\alpha}$. Therefore (19) becomes

$$\frac{1}{M}\sum_{i=1}^{M}\|e_{t+1,i}\|^2\leq\frac{\beta}{1-\alpha}=\frac{2(2-\delta)G^2}{\delta^2}. \qquad (21)$$

Let us now consider the term $\|\tilde{e}_{t+1}\|^2$ of (9). With $M_t=|S_t|$, we have the following

$$\begin{aligned}\|\tilde{e}_{t+1}\|^2&=\|\tilde{c}_t\mathsf{sign}_{\tilde{\beta}_t}(\tilde{p}_t)-\tilde{p}_t\|^2\\&\leq(1-\delta_{\tilde{\beta}_t})\|\tilde{p}_t\|^2\\&=(1-\delta_{\tilde{\beta}_t})\left\|\frac{1}{M_t}\sum_{i\in S_t}c_t'\mathsf{sign}_{\beta_{t,i}}(p_{t,i})+\gamma\tilde{e}_t\right\|^2\\&\leq(1-\delta_{\tilde{\beta}_t})\gamma^2(1+\lambda)\|\tilde{e}_t\|^2\\&\quad+(1-\delta_{\tilde{\beta}_t})(1+1/\lambda)\left\|\frac{1}{M_t}\sum_{i\in S_t}c_t'\mathsf{sign}_{\beta_{t,i}}(p_{t,i})\right\|^2\\&\leq(1-\tilde{\delta})(1+\lambda)\|\tilde{e}_t\|^2\\&\quad+(1-\tilde{\delta})(1+1/\lambda)\left\|\frac{1}{M_t}\sum_{i\in S_t}c_t'\mathsf{sign}_{\beta_{t,i}}(p_{t,i})\right\|^2\end{aligned}$$

where the second inequality is by Young inequality for any $\lambda>0$, and the last inequality is by $\gamma\leq 1$ and $\tilde{\delta}=\min\{\delta_{\tilde{\beta}_t}\}$. We additionally have

$$\begin{aligned}&\left\|\frac{1}{M_t}\sum_{i\in S_t}c_t'\mathsf{sign}_{\beta_{t,i}}(p_{t,i})\right\|^2\\&\leq\frac{1}{M_t}\sum_{i\in S_t}\|c_t'\mathsf{sign}_{\beta_{t,i}}(p_{t,i})\|^2\\&\leq\frac{1}{M_t}\sum_{i\in S_t}\left(2\|c_t'\mathsf{sign}_{\beta_{t,i}}(p_{t,i})-p_{t,i}\|^2+2\|p_{t,i}\|^2\right)\\&\leq\frac{1}{M_t}\sum_{i\in S_t}\left(2(1-\delta_{\beta_{t,i}})\|p_{t,i}\|^2+2\|p_{t,i}\|^2\right) \qquad (22)\\&=\frac{1}{M_t}\sum_{i\in S_t}2(2-\delta_{\beta_{t,i}})\|p_{t,i}\|^2\\&\leq 2(2-\delta)\frac{1}{M_t}\sum_{i\in S_t}\|p_{t,i}\|^2, \qquad (23)\end{aligned}$$

where (22) is by Lemma 1 of [27], and (23) is by $\delta\leq\min\{\delta_{\beta_{t,i}}\}$. Therefore

$$\begin{aligned}&\|\tilde{e}_{t+1}\|^2\\&\leq(1-\tilde{\delta})(1+\lambda)\|\tilde{e}_t\|^2\\&\quad+2(2-\delta)(1-\tilde{\delta})(1+1/\lambda)\frac{1}{M_t}\sum_{i\in S_t}\|p_{t,i}\|^2\\&=(1-\tilde{\delta})(1+\lambda)\|\tilde{e}_t\|^2\\&\quad+2(2-\delta)(1-\tilde{\delta})(1+1/\lambda)\frac{M}{M_t}\frac{1}{M}\sum_{i\in S_t}\|p_{t,i}\|^2.\end{aligned}$$

Since (21) holds for all t, we get $\frac{1}{M}\sum_{i=1}^{M}\|e_{t,i}\|^2\leq\frac{\beta}{1-\alpha}$. Moreover, recall that $(1-\delta)(1+\lambda)=\alpha$ from (18) and $\beta=\frac{(2-\delta)G^2}{\delta}$ from (20). Therefore the upper bound for (16) is

as follows

$$(16) \leq \alpha \cdot \frac{\beta}{1-\alpha} + \beta = \frac{\beta}{1-\alpha} = \frac{2(2-\delta)G^2}{\delta^2}.$$

This inequality and (12) yield

$$\frac{(1-\delta)}{M}\sum_{i\in S_t}\|p_{t,i}\|^2 + \frac{1}{M}\sum_{i\notin S_t}\|p_{t,i}\|^2 \leq \frac{2(2-\delta)G^2}{\delta^2}$$

which implies

$$\frac{1}{M}\sum_{i\in S_t}\|p_{t,i}\|^2 \leq \frac{2(2-\delta)G^2}{\delta^2(1-\delta)}.$$

Therefore

$$\|\tilde{e}_{t+1}\|^2 \leq (1-\tilde{\delta})(1+\lambda)\|\tilde{e}_t\|^2 + \frac{4(2-\delta)^2(1-\tilde{\delta})(1+1/\lambda)G^2M}{(1-\delta)\delta^2M_t}.$$

Hereafter, choosing $\lambda = \frac{\tilde{\delta}}{2(1-\tilde{\delta})}$, we obtain

$$(1-\tilde{\delta})(1+\lambda) = 1 - \frac{\tilde{\delta}}{2}1 + 1/\lambda = \frac{2-\tilde{\delta}}{\tilde{\delta}}.$$

Therefore

$$\|\tilde{e}_{t+1}\|^2 \leq \left(1-\frac{\tilde{\delta}}{2}\right)\|\tilde{e}_t\|^2 + \frac{4(1-\tilde{\delta})(2-\tilde{\delta})(2-\delta)^2G^2M}{\tilde{\delta}(1-\delta)\delta^2M_t}. \quad (24)$$

Note that inequality (24) is of the form

$$a_{t+1} \leq \alpha a_t + \beta, \quad (25)$$

where

$$a_{t+1} = \|\tilde{e}_{t+1}\|^2$$
$$\alpha = 1 - \frac{\tilde{\delta}}{2}$$
$$\beta = \frac{4(1-\tilde{\delta})(2-\tilde{\delta})(2-\delta)^2G^2M}{\tilde{\delta}(1-\delta)\delta^2M_t}.$$

Applying Lemma 1 of [32], we obtain

$$\|\tilde{e}_{t+1}\|^2 \leq \beta\sum_{j=0}^{t}\alpha^j \leq \frac{8(1-\tilde{\delta})(2-\tilde{\delta})(2-\delta)^2G^2M}{(\tilde{\delta})^2(1-\delta)\delta^2M_t} \quad (26)$$

where (26) is by the fact that

$$\sum_{j=0}^{t}\alpha^j \leq \sum_{j\geq 0}\alpha^j = \frac{1}{1-\alpha} = \frac{2}{\tilde{\delta}}.$$

Substituting (21) and (26) into (9) gives us

$$\left\|\tilde{e}_{t+1} + \frac{1}{M}\sum_{i=1}^{M}e_{t+1,i}\right\|^2 \leq \frac{4(2-\delta)G^2}{\delta^2} + \frac{16(2-\tilde{\delta})(1-\tilde{\delta})(2-\delta)(2-\delta)G^2M}{(\tilde{\delta})^2(1-\delta)\delta^2M_t} = \frac{4(2-\delta)G^2}{\delta^2}\left(1 + \frac{4(2-\tilde{\delta})(1-\tilde{\delta})(2-\delta)M}{(\tilde{\delta})^2(1-\delta)M_t}\right)$$

Let

$$U_t = \frac{4(2-\delta)}{\delta^2}\left(1 + \frac{4(2-\tilde{\delta})(1-\tilde{\delta})(2-\delta)M}{(\tilde{\delta})^2(1-\delta)M_t}\right)$$

and $U = \max\{U_0, \ldots, U_{T-1}\}$, we obtain

$$\left\|\tilde{e}_{t+1} + \frac{1}{M}\sum_{i=1}^{M}e_{t+1,i}\right\|^2 \leq G^2U$$

and the claim follows. ■

REFERENCES

[1] J. Dean, G. Corrado, R. Monga, K. Chen, M. Devin, Q. V. Le, M. Z. Mao, M. Ranzato, A. W. Senior, P. A. Tucker, K. Yang, and A. Y. Ng, "Large scale distributed deep networks," in *Proc. 26th Annu. Conf. Neural Inf. Process. Syst.*, 2012, pp. 1232–1240. [Online]. Available: http://papers.nips.cc/paper/4687-large-scale-distributed-deep-networks

[2] B. Recht, C. Re, S. Wright, and F. Niu, "Hogwild! A lock-free approach to parallelizing stochastic gradient descent," in *Advances in Neural Information Processing Systems*, vol. 24, J. Shawe-Taylor, R. S. Zemel, P. L. Bartlett, F. Pereira, and K. Q. Weinberger, Eds. Red Hook, NY, USA: Curran Associates, 2011, pp. 693–701. [Online]. Available: http://papers.nips.cc/paper/4390-hogwild-a-lock-free-approach-to-parallelizing-stochastic-gradient-descent.pdf

[3] J. Dean and L. A. Barroso, "The tail at scale," *Commun. ACM*, vol. 56, no. 2, pp. 74–80, Feb. 2013. [Online]. Available: http://cacm.acm.org/magazines/2013/2/160173-the-tail-at-scale/fulltext

[4] S. Zheng, Z. Huang, and J. T. Kwok, "Communication-efficient distributed blockwise momentum SGD with error-feedback," in *Proc. Adv. Neural Inf. Process. Syst., Annu. Conf. Neural Inf. Process. Syst. (NeurIPS)*, vol. 32, 2019, pp. 11446–11456. [Online]. Available: http://papers.nips.cc/paper/9321-communication-efficient-distributed-blockwise-momentum-sgd-with-error-feedback

[5] H. Tang, C. Yu, X. Lian, T. Zhang, and J. Liu, "DoubleSqueeze: Parallel stochastic gradient descent with double-pass error-compensated compression," in *Proc. 36th Int. Conf. Mach. Learn.*, vol. 97, K. Chaudhuri and R. Salakhutdinov, Eds., 2019, pp. 6155–6165. [Online]. Available: http://proceedings.mlr.press/v97/tang19d.html

[6] P. Kairouz et al., "Advances and open problems in federated learning," *CoRR*, vol. abs/1912.04977, p. 121, Dec. 2019.

[7] D. Basu, D. Data, C. Karakus, and S. N. Diggavi, "Qsparse-local-SGD: Distributed SGD with quantization, sparsification and local computations," in *Proc. Adv. Neural Inf. Process. Syst., Annu. Conf. Neural Inf. Process. Syst. (NeurIPS)*, vol. 32, 2019, pp. 14668–14679. [Online]. Available: http://papers.nips.cc/paper/9610-qsparse-local-sgd-distributed-sgd-with-quantization-sparsification-and-local-computations

[8] C. Karakus, Y. Sun, S. Diggavi, and W. Yin, "Redundancy techniques for straggler mitigation in distributed optimization and learning," *J. Mach. Learn. Res.*, vol. 20, no. 72, pp. 1–47, 2019. [Online]. Available: http://jmlr.org/papers/v20/18-148.html

[9] D. Yin, Y. Chen, R. Kannan, and P. Bartlett, "Byzantine-robust distributed learning: Towards optimal statistical rates," in *Proc. 35th Int. Conf. Mach. Learn.*, vol. 80, J. Dy and A. Krause, Eds., 2018, pp. 5650–5659. [Online]. Available: http://proceedings.mlr.press/v80/yin18a.html

[10] E. M. El Mhamdi, R. Guerraoui, and S. Rouault, "The hidden vulnerability of distributed learning in Byzantium," in *Proc. 35th Int. Conf. Mach. Learn.*, vol. 80, J. Dy and A. Krause, Eds., Stockholm Sweden, Jul. 2018, pp. 3521–3530.

[11] P. Richtárik, I. Sokolov, E. Gasanov, I. Fatkhullin, Z. Li, and E. Gorbunov, "3PC: Three point compressors for communication-efficient distributed training and a better theory for lazy aggregation," in *Proc. Int. Conf. Mach. Learn. (ICML)*, vol. 162, K. Chaudhuri, S. Jegelka, L. Song, C. Szepesvári, G. Niu, and S. Sabato, Eds., Baltimore, MD, USA, Jul. 2022, pp. 18596–18648. [Online]. Available: https://proceedings.mlr.press/v162/richtarik22a.html

[12] E. Gasanov, A. Khaled, S. Horváth, and P. Richtárik, "FLIX: A simple and communication-efficient alternative to local methods in federated learning," in *Proc. Int. Conf. Artif. Intell. Statist. (AISTATS)*, vol. 151, G. Camps-Valls, F. J. R. Ruiz, and I. Valera, Eds., Mar. 2022, pp. 11374–11421. [Online]. Available: https://proceedings.mlr.press/v151/gasanov22a.html

[13] K. Gardner, S. Zbarsky, S. Doroudi, M. Harchol-Balter, and E. Hyytiä, "Reducing latency via redundant requests: Exact analysis," in *Proc. ACM SIGMETRICS Int. Conf. Meas. Modeling Comput. Syst.*, B. Lin, J. J. Xu, S. Sengupta, and D. Shah, Eds., Portland, OR, USA, Jun. 2015, pp. 347–360, doi: 10.1145/2745844.2745873.

[14] G. Ananthanarayanan, A. Ghodsi, S. Shenker, and I. Stoica, "Effective straggler mitigation: Attack of the clones," in *Proc. 10th USENIX Symp. Netw. Syst. Design Implement. (NSDI)*, N. Feamster and J. C. Mogul, Eds. Lombard, IL, USA: USENIX Association, Apr. 2013, pp. 185–198. [Online]. Available: https://www.usenix.org/conference/nsdi13/technical-sessions/presentation/ananthanarayanan

[15] N. B. Shah, K. Lee, and K. Ramchandran, "When do redundant requests reduce latency?" *IEEE Trans. Commun.*, vol. 64, no. 2, pp. 715–722, Feb. 2016, doi: 10.1109/TCOMM.2015.2506161.

[16] D. Wang, G. Joshi, and G. Wornell, "Using straggler replication to reduce latency in large-scale parallel computing," *ACM SIGMETRICS Perform. Eval. Rev.*, vol. 43, no. 3, pp. 7–11, Nov. 2015, doi: 10.1145/2847220.2847223.

[17] N. J. Yadwadkar, B. Hariharan, E. J. Gonzalez, and R. H. Katz, "Multi-task learning for straggler avoiding predictive job scheduling," *J. Mach. Learn. Res.*, vol. 17, pp. 106:1–106:37, Jan. 2016. [Online]. Available: http://jmlr.org/papers/v17/15-149.html

[18] A. Agarwal and J. C. Duchi, "Distributed delayed stochastic optimization," in *Proc. Adv. Neural Inf. Process. Syst., 25th Annu. Conf. Neural Inf. Process. Syst.*, vol. 24, J. Shawe-Taylor, R. S. Zemel, P. L. Bartlett, F. C. N. Pereira, and K. Q. Weinberger, Eds., Granada, Spain, Dec. 2011, pp. 873–881. [Online]. Available: http://papers.nips.cc/paper/4247-distributed-delayed-stochastic-optimization

[19] M. Li, D. G. Andersen, J. W. Park, A. J. Smola, A. Ahmed, V. Josifovski, J. Long, E. J. Shekita, and B. Su, "Scaling distributed machine learning with the parameter server," in *Proc. 11th USENIX Symp. Operating Syst. Design Implement. (OSDI)*, J. Flinn and H. Levy, Eds. Broomfield, CO, USA: USENIX Association, Oct. 2014, pp. 583–598. [Online]. Available: https://www.usenix.org/conference/osdi14/technical-sessions/presentation/li_mu

[20] T. Vogels, S. P. Karimireddy, and M. Jaggi, "PowerSGD: Practical low-rank gradient compression for distributed optimization," in *Advances in Neural Information Processing Systems*, vol. 32, H. Wallach, H. Larochelle, A. Beygelzimer, F. d'Alché-Buc, E. Fox, and R. Garnett, Eds. Red Hook, NY, USA: Curran Associates, 2019, pp. 14236–14245. [Online]. Available: http://papers.nips.cc/paper/9571-powersgd-practical-low-rank-gradient-compression-for-distributed-optimization.pdf

[21] T. Chen, G. B. Giannakis, T. Sun, and W. Yin, "LAG: Lazily aggregated gradient for communication-efficient distributed learning," in *Proc. Adv. Neural Inf. Process. Syst., Annu. Conf. Neural Inf. Process. Syst. (NeurIPS)*, vol. 31, S. Bengio, H. M. Wallach, H. Larochelle, K. Grauman, N. Cesa-Bianchi, and R. Garnett, Eds., Montreal, QC, Canada, Dec. 2018, pp. 5055–5065. [Online]. Available: https://proceedings.neurips.cc/paper/2018/hash/feecee9f1643651799ede2740927317a-Abstract.html

[22] H. S. Ghadikolaei, S. U. Stich, and M. Jaggi, "LENA: Communication-efficient distributed learning with self-triggered gradient uploads," in *Proc. 24th Int. Conf. Artif. Intell. Statist. (AISTATS)*, vol. 130, A. Banerjee and K. Fukumizu, Eds., Apr. 2021, pp. 3943–3951. [Online]. Available: http://proceedings.mlr.press/v130/shokri-ghadikolaei21a.html

[23] K. Mishchenko, B. Wang, D. Kovalev, and P. Richtárik, "IntSGD: Adaptive floatless compression of stochastic gradients," in *Proc. Int. Conf. Learn. Represent.*, 2022, pp. 1–27. [Online]. Available: https://openreview.net/forum?id=pFyXqxChZc

[24] A. Xu and H. Huang, "Detached error feedback for distributed SGD with random sparsification," in *Proc. 39th Int. Conf. Mach. Learn.*, vol. 162, K. Chaudhuri, S. Jegelka, L. Song, C. Szepesvari, G. Niu, and S. Sabato, Eds., Jul. 2022, pp. 24550–24575. [Online]. Available: https://proceedings.mlr.press/v162/xu22c.html

[25] S. Horváth, D. Kovalev, K. Mishchenko, P. Richtárik, and S. U. Stich, "Stochastic distributed learning with gradient quantization and double-variance reduction," *Optim. Methods Softw.*, vol. 38, no. 1, pp. 91–106, 2023, doi: 10.1080/10556788.2022.2117355.

[26] J. Bernstein, J. Zhao, K. Azizzadenesheli, and A. Anandkumar, "signSGD with majority vote is communication efficient and fault tolerant," in *Proc. 7th Int. Conf. Learn. Represent. (ICLR)*, 2019, pp. 1–20. [Online]. Available: https://openreview.net/forum?id=BJxhijAcY7

[27] T. Le Phong and T. T. Phuong, "Distributed signSGD with improved accuracy and network-fault tolerance," *IEEE Access*, vol. 8, pp. 191839–191849, 2020, doi: 10.1109/ACCESS.2020.3032637.

[28] F. Sattler, S. Wiedemann, K.-R. Müller, and W. Samek, "Robust and communication-efficient federated learning from non-i.i.d. data," *IEEE Trans. Neural Netw. Learn. Syst.*, vol. 31, no. 9, pp. 3400–3413, Sep. 2020.

[29] D. Rothchild, A. Panda, E. Ullah, N. Ivkin, I. Stoica, V. Braverman, J. Gonzalez, and R. Arora, "FetchSGD: Communication-efficient federated learning with sketching," in *Proc. 37th Int. Conf. Mach. Learn. (ICML)*, vol. 119, Jul. 2020, pp. 8253–8265. [Online]. Available: http://proceedings.mlr.press/v119/rothchild20a.html

[30] J. Wang and G. Joshi, "Cooperative SGD: A unified framework for the design and analysis of local-update SGD algorithms," *J. Mach. Learn. Res.*, vol. 22, no. 213, pp. 1–50, 2021. [Online]. Available: http://jmlr.org/papers/v22/20-147.html

[31] F. Haddadpour, M. M. Kamani, A. Mokhtari, and M. Mahdavi, "Federated learning with compression: Unified analysis and sharp guarantees," in *Proc. 24th Int. Conf. Artif. Intell. Statist. (AISTATS)*, vol. 130, A. Banerjee and K. Fukumizu, Eds., Apr. 2021, pp. 2350–2358. [Online]. Available: http://proceedings.mlr.press/v130/haddadpour21a.html

[32] T. T. Phuong and L. T. Phong, "Distributed SGD with flexible gradient compression," *IEEE Access*, vol. 8, pp. 64707–64717, 2020, doi: 10.1109/ACCESS.2020.2984633.

[33] S. P. Karimireddy, Q. Rebjock, S. U. Stich, and M. Jaggi, "Error feedback fixes signSGD and other gradient compression schemes," in *Proc. 36th Int. Conf. Mach. Learn. (ICML)*, 2019, pp. 3252–3261. [Online]. Available: http://proceedings.mlr.press/v97/karimireddy19a.html

[34] G. Yan, T. Li, S.-L. Huang, T. Lan, and L. Song, "AC-SGD: Adaptively compressed SGD for communication-efficient distributed learning," *IEEE J. Sel. Areas Commun.*, vol. 40, no. 9, pp. 2678–2693, Sep. 2022.

[35] A. Sapio, M. Canini, C. Ho, J. Nelson, P. Kalnis, C. Kim, A. Krishnamurthy, M. Moshref, D. R. K. Ports, and P. Richtárik, "Scaling distributed machine learning with in-network aggregation," in *Proc. 18th USENIX Symp. Netw. Syst. Design Implement. (NSDI)*, J. Mickens and R. Teixeira, Eds. Berkeley, CA, USA: USENIX Association, Apr. 2021, pp. 785–808. [Online]. Available: https://www.usenix.org/conference/nsdi21/presentation/sapio

[36] J. Chen, R. Monga, S. Bengio, and R. Jozefowicz, "Revisiting distributed synchronous SGD," in *Proc. Int. Conf. Learn. Represent. Workshop Track*, 2016, pp. 1–10.

[37] S. Dutta, G. Joshi, S. Ghosh, P. Dube, and P. Nagpurkar, "Slow and stale gradients can win the race: Error-runtime trade-offs in distributed SGD," in *Proc. Int. Conf. Artif. Intell. Statist. (AISTATS)* Playa Blanca, Spain Apr. 2018, pp. 803–812. [Online]. Available: http://proceedings.mlr.press/v84/dutta18a.html

[38] A. Nedic and A. Olshevsky, "Distributed optimization over time-varying directed graphs," *IEEE Trans. Autom. Control*, vol. 60, no. 3, pp. 601–615, Mar. 2015, doi: 10.1109/TAC.2014.2364096.

[39] A. Nedić, A. Olshevsky, and W. Shi, "Achieving geometric convergence for distributed optimization over time-varying graphs," *SIAM J. Optim.*, vol. 27, no. 4, pp. 2597–2633, Jan. 2017, doi: 10.1137/16M1084316.

[40] O. Russakovsky, J. Deng, H. Su, J. Krause, S. Satheesh, S. Ma, Z. Huang, A. Karpathy, A. Khosla, M. Bernstein, A. C. Berg, and L. Fei-Fei, "ImageNet large scale visual recognition challenge," *Int. J. Comput. Vis.*, vol. 115, no. 3, pp. 211–252, Dec. 2015.

[41] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," *CoRR*, vol. abs/1512.03385, pp. 1–12, Dec. 2015.
[42] Z. Huang. Accessed: Aug. 15, 2023. [Online]. Available: https://github.com/ZiyueHuang/dist-ef-sgdm/tree/master/imagenet

TRAN THI PHUONG received the Ph.D. degree in mathematics from Meiji University, in 2012. She has published around 20 research articles in academic journals, covering topics within the fields of mathematics and machine learning. She has focused on communication efficiency, robustness, mathematical convergence, and privacy in machine learning algorithms. She seeks to improve communication efficiency in distributed machine learning algorithms and develop robust algorithms that can function in noisy communication environments. She identifies the mathematical principles that underpin convergence in machine learning models. She has also developed techniques that enable machine learning models to maintain data privacy when handling sensitive data. Her research interests include the intersection of mathematics and machine learning and the performance and capabilities of machine learning algorithms while ensuring data privacy and security.

LE TRIEU PHONG received the Ph.D. degree from the Tokyo Institute of Technology, in 2009. He has authored over 40 articles in the fields of machine learning and cryptography, which have been published in both conference proceedings and academic journals. In machine learning, he has focused on creating new algorithms capable of effectively processing vast amounts of data, while his cryptography research has centered on devising techniques for secure data transmission and storage. The goal of his research is to enhance the performance and capabilities of machine learning algorithms, while simultaneously prioritizing the protection of data privacy and security.

KAZUHIDE FUKUSHIMA received the M.E. degree in information engineering and the Ph.D. degree in engineering from Kyushu University, Japan, in 2004 and 2009, respectively. He joined KDDI and has been engaged in the research on post-quantum cryptography, cryptographic protocols, and identification technologies. He is currently the Senior Manager of the Information Security Laboratory, KDDI Research Inc. He is a Senior Member of the Institute of Electronics, Information and Communication Engineers (IEICE) and a member of the Information Processing Society of Japan (IPSJ). He received the IEICE Young Engineer Award, in 2012. He served as an Editor for *IEICE Transactions on Fundamentals of Electronics Communications and Computer Sciences*, from 2015 to 2017, and the Director of the General Affairs of IEICE Engineering Science Society, from 2019 to 2021.