Over-the-Air Machine Learning at the Wireless Edge

Mohammad Mohammadi Amiri and Deniz Gündüz
Electrical and Electronic Engineering Department, Imperial College London, London SW7 2BT, U.K.
Email: {m.mohammadi-amiri15, d.gunduz}@imperial.ac.uk

***Abstract*—We study distributed machine learning at the wireless edge, where limited power devices (*workers*) with local datasets implement distributed stochastic gradient descent (DSGD) over-the-air with the help of a remote parameter server (PS). We consider a bandwidth-limited fading multiple access channel (MAC) from the workers to the PS for communicating the local gradient estimates. Motivated by the additive nature of the wireless MAC, we study *analog* transmission of low-dimensional gradient estimates while accumulating error from previous iterations. We also design an opportunistic worker scheduling scheme to align the received gradient vectors at the PS in an efficient manner. Numerical results show that the proposed DSGD algorithm converges much faster than the state-of-the-art, while also providing a significantly higher accuracy.**

I. Introduction

Many emerging technologies require collection and processing of massive amounts of data. Current trend is to employ centralized algorithms, where a powerful machine learning technique, often a neural network, is trained on a massive dataset collected and offloaded by edge devices. However, in the case of wireless edge devices, sending such massive amounts of data to a central processor in a reliable manner may be too costly in terms of energy and bandwidth, and may not meet the latency and privacy requirements of the underlying application. With the current technology, communication is typically more costly than processing; thus, a much more desirable approach is to develop distributed machine learning techniques that can utilize the local processing capabilities of edge devices, and require only limited amount of communications. In this paper, we consider machine learning at the wireless edge, where distributed wireless devices (mobile phones, tablets, IoT devices) with local datasets help jointly train a learning model with the help of a *parameter server* (PS), to which they connect through a shared wireless channel.

Machine learning problems often involve the minimization of the empirical loss function $F(\boldsymbol{\theta}) = \frac{1}{D}\sum_{i=1}^{D} f(\boldsymbol{\theta}, \boldsymbol{u}_i)$, where $\boldsymbol{\theta} \in \mathbb{C}^d$ denotes the model parameters to be optimized, $\boldsymbol{u}_i$ is the i-th training data sample, $i \in [D] \triangleq \{1, \dots, D\}$, and $f(\cdot)$ is the loss function defined by the learning model. Minimization of $F(\boldsymbol{\theta})$ is typically carried out through iterative stochastic gradient descent (SGD), in which the model parameters at iteration t, $\boldsymbol{\theta}_t$, are updated with a stochastic gradient $\boldsymbol{\theta}_{t+1} = \boldsymbol{\theta}_t - \eta_t \boldsymbol{g}(\boldsymbol{\theta}_t)$, which satisfies $\mathbb{E}[\boldsymbol{g}(\boldsymbol{\theta}_t)] = \nabla F(\boldsymbol{\theta}_t)$, where η_t is the learning rate. SGD also allows parallelization when the dataset is distributed across multiple computation servers, called the *workers*. In distributed SGD (DSGD), at each iteration, worker m computes a gradient vector based on the global parameter vector with respect to its local dataset, denoted by $\mathcal{B}_m$, and sends the result to the PS, which updates the global parameter vector according to

$$\boldsymbol{\theta}_{t+1} = \boldsymbol{\theta}_t - \eta_t \frac{1}{M} \sum\nolimits_{m=1}^{M} \boldsymbol{g}_m(\boldsymbol{\theta}_t), \qquad (1)$$

where M denotes the number of workers, and $\boldsymbol{g}_m(\boldsymbol{\theta}_t) \triangleq \frac{1}{|\mathcal{B}_m|}\sum_{\boldsymbol{u}_i \in \mathcal{B}_m} \nabla f(\boldsymbol{\theta}_t, \boldsymbol{u}_i)$, $m \in [M]$. While parallelism allows exploiting the computing power of M workers, communications becomes the main bottleneck [1]–[5]. This is an even bigger hurdle in wireless edge learning due to stringent bandwidth and energy constraints.

Numerous studies have focused on communication-efficient DSGD schemes, where three main approaches are employed; namely, *quantization* [1]–[3], *sparsification* [4], [6], [7], and *local updates* [8], [9]. However, these works ignore the physical aspects of the underlying communication channel, and focus on reducing the amount of data sent by each worker to the PS.

In this paper, we consider DSGD *over-the-air*; that is, we consider a wireless shared medium from the workers to the PS, and treat each iteration of the DSGD algorithm as a distributed over-the-air computation problem. We focus on analog transmission from the workers to the PS [10]–[12], which is shown in [12] to outperform the digital transmission approach significantly for distributed learning applications thanks to the signal-superposition property of the wireless MAC. The authors in [10] study a distributed learning problem over a fading MAC, where each entry of a gradient vector is scheduled for transmission depending on its corresponding channel condition. A SIMO wireless MAC with beamforming is considered in [11] for distributed learning, where the goal is to maximize the number of workers scheduled for transmission with an acceptable quality for the retrieved signal.

Here, we extend our previous work in [12] that focused on DSGD over a Gaussian MAC to a fading MAC. We will show that the proposed scheme, in which each worker compresses its gradient estimate to a low-dimensional vector while accumulating the error, outperforms the one studied in [10] substantially.

This work was partially supported by the European Research Council (ERC) through project BEACON (No. 725731), and by the European Union's Horizon 2020 Research and Innovation Programme through project SCAVENGE (No. 675891).

Notations: $\mathbb{R}$ and $\mathbb{C}$ represent the sets of real and complex values, respectively. For two vectors $\boldsymbol{x}$ and $\boldsymbol{y}$ with the same dimension, $\boldsymbol{x} \cdot \boldsymbol{y}$ returns their inner product. We denote a zero-mean normal distribution with variance σ^2 by $\mathcal{N}\left(0, \sigma^2\right)$, and $\mathcal{CN}\left(0, \sigma^2\right)$ represents a complex normal distribution with real and imaginary terms each distributed according to $\mathcal{N}\left(0, \sigma^2/2\right)$. We let $[i] \triangleq \{1, \dots, i\}$. We denote the cardinality of set $\mathcal{A}$ by $|\mathcal{A}|$, and l_2 norm of vector $\boldsymbol{x}$ by $\|\boldsymbol{x}\|_2$.

II. System Model

We consider distributed edge learning, where M nodes (*workers*), connected to a remote PS through a wireless fading MAC, employ SGD with the help of the PS. We denote the set of data samples available at worker m by $\mathcal{B}_m$, and the stochastic gradient computed by worker m with respect to local data samples by $\boldsymbol{g}_m\left(\boldsymbol{\theta}_t\right) \in \mathbb{C}^d$, $m \in [M]$. At the t-th iteration of DSGD algorithm in (1), the local gradient estimates of the workers are sent to the PS over a wireless MAC using s subchannels for a total of N time slots, where $s \le d$.[1] We denote the length-s channel input vector transmitted by worker m at the n-th time slot of the t-th iteration of the DSGD by $\boldsymbol{x}_m^n(t) = [x_{m,1}^n(t) \cdots x_{m,s}^n(t)]^T \in \mathbb{C}^s$. The i-th entry of channel output $\boldsymbol{y}^n(t) \in \mathbb{C}^s$ received by the PS at the t-th iteration and n-th time slot, $n \in [N]$, is given by

$$y_i^n(t) = \sum\nolimits_{m \in \mathcal{M}_i^n(t)} h_{m,i}^n(t) x_{m,i}^n(t) + z_i^n(t), \quad i \in [s], \tag{2}$$

where $\mathcal{M}_i^n(t) \subset [M]$, $h_{m,i}^n(t) \in \mathbb{C}$ is the i-th entry of the vector of channel gains $\boldsymbol{h}_m^n(t)$ from worker m to the PS, and is assumed to be independent and identically distributed (i.i.d.) according to $\mathcal{CN}(0, \sigma_m^2)$, e.g., Rayleigh fading, and $z_i^n(t) \in \mathbb{C}$ is the i-th entry of complex Gaussian noise vector $\boldsymbol{z}^n(t)$ and is i.i.d. according to $\mathcal{CN}\left(0, 1\right)$. The channel input vector of worker m at the n-th time slot of iteration t, $n \in [N]$, is a function of the channel gains $\boldsymbol{h}_m^n(t)$, current parameter vector $\boldsymbol{\theta}_t$, the local dataset $\mathcal{B}_m$, and the current gradient estimate at worker m, $\boldsymbol{g}_m\left(\boldsymbol{\theta}_t\right)$, $m \in [M]$. We assume that, at each time slot, the channel state information (CSI) is known by the workers and the PS. For a total of T iterations of the DSGD algorithm, the following total average transmit power constraint is imposed at worker m:

$$\frac{1}{NT} \sum\nolimits_{t=1}^{T} \sum\nolimits_{n=1}^{N} \mathbb{E}\left[||\boldsymbol{x}_m^n(t)||_2^2 \right] \le \bar{P}, \quad \forall m \in [m], \tag{3}$$

where the expectation is taken over the randomness of the channel gains.

The goal is to recover $\frac{1}{M} \sum_{m=1}^{M} \boldsymbol{g}_m\left(\boldsymbol{\theta}_t\right)$ at the PS, which then updates the model parameter as in (1) after N time slots. However, due to the pre-processing performed at each worker and the distortion caused by the wireless channel, the PS uses a noisy estimate to update the model parameter. Having defined $\boldsymbol{y}(t) \triangleq [\boldsymbol{y}^1(t)^T \cdots \boldsymbol{y}^N(t)^T]^T$, we have $\boldsymbol{\theta}_{t+1} = \phi(\boldsymbol{\theta}_t, \boldsymbol{y}(t))$ for some update function $\phi : \mathbb{C}^d \times \mathbb{C}^{Ns} \to \mathbb{C}^d$. The updated model parameter is then multicast to the workers by the PS through an error-free shared link, so the workers receive a consistent parameter vector for their computations in the next iteration.

[1]In many machine learning applications, d is extremely large, e.g., the 50-layer ResNet network has ~ 26 million weight parameters, whereas the channel bandwidth, measured by parameter s, is small due to the bandwidth and latency limitations; for example 1 LTE frame of 5MHz bandwidth and duration 10ms can carry only 6000 complex symbols.

We note that the goal of the workers is to transmit their local gradient estimates to the PS over the wireless channel, which is a joint source-channel coding problem. Moreover, the PS is not interested in the individual estimates of the workers, but in their average; hence, we have a distributed computation problem. We will consider an analog transmission approach, where the gradients are transmitted simultaneously over the wireless MAC in an uncoded fashion.

III. Analog DSGD

Analog DSGD is motivated by the fact that the PS is only interested in the average of the gradient vectors, and the underlying wireless MAC can provide the sum of the gradients if they are sent without any coding. We first present a generalization of the analog computation approach introduced in [10], referred to as the *entry-wise scheduled analog DSGD (ESA-DSGD)* scheme, and then propose our scheme, built upon our previous work [12], referred to as the *compressed worker-wise scheduled analog DSGD (CWSA-DSGD)* scheme.

A. ESA-DSGD

With the ESA-DSGD scheme in [10], each worker sends its gradient estimate entirely after applying power allocation to satisfy the average power constraint. At the t-th iteration of the DSGD, worker m, $m \in [M]$, transmits its local gradient estimate $\boldsymbol{g}_m\left(\boldsymbol{\theta}_t\right) \in \mathbb{C}^d$ over $N = \lceil d/s \rceil$ time slots of the available s subchannels. We define $\boldsymbol{g}_m^n\left(\boldsymbol{\theta}_t\right) \triangleq [g_{m,(n-1)s+1}\left(\boldsymbol{\theta}_t\right) \cdots g_{m,ns}\left(\boldsymbol{\theta}_t\right)]^T$, $n \in [N]$, $m \in [M]$, where $g_{m,i}\left(\boldsymbol{\theta}_t\right)$ is the i-th entry of $\boldsymbol{g}_m\left(\boldsymbol{\theta}_t\right)$, and we zero-pad $\boldsymbol{g}_m\left(\boldsymbol{\theta}_t\right)$ to have dimension Ns. At the n-th time slot of the t-th iteration of the DSGD, worker m, $m \in [M]$, sends $\boldsymbol{x}_m^n\left(t\right) = \boldsymbol{\beta}_m^n(t) \cdot \boldsymbol{g}_m^n\left(\boldsymbol{\theta}_t\right)$, where $\boldsymbol{\beta}_m^n(t) \in \mathbb{C}^s$ is the power allocation vector, which is set to satisfy the average power constraint. Thus, after N time slots, each worker sends its gradient estimate, of dimension d, entirely. The i-th entry of the power allocation vector $\boldsymbol{\beta}_m^n(t)$ is set as follows:

$$\beta_{m,i}^n(t) = \begin{cases} \frac{\lambda(t)}{h_{m,i}^n(t)}, & \text{if } \left|h_{m,i}^n(t)\right|^2 \ge h_{\text{th}}^{\text{e}}(t), \\ 0, & \text{otherwise}, \end{cases} \tag{4}$$

for some $\lambda(t) \in \mathbb{R}$, where $h_{\text{th}}^{\text{e}}(t)$ is chosen to satisfy the average power constraint. According to (4), each entry of a gradient vector is transmitted if its corresponding channel gain is over a threshold. The set of workers selected to transmit the i-th entry of the channel input vector at the n-th time slot is given by, $i \in [s]$, $n \in [N]$,

$$\mathcal{M}_i^n(t) = \left\{ m \in [M] : \left|h_{m,i}^n(t)\right|^2 \ge h_{\text{th}}^{\text{e}}(t) \right\}. \tag{5}$$

By substituting $\boldsymbol{x}_m^n\left(\boldsymbol{\theta}_t\right)$ and $\boldsymbol{\beta}_m^n(t)$ into (2), it follows that, for $i \in [s]$, $n \in [N]$,

$$y_i^n(t) = \lambda(t) \sum\nolimits_{m \in \mathcal{M}_i^n(t)} g_{m,(n-1)s+i}(\boldsymbol{\theta}_t) + z_i^n(t). \tag{6}$$

The PS has perfect CSI, and hence, knows set $\mathcal{M}_i^n(t)$. Its goal is to recover $\frac{1}{|\mathcal{M}_i^n(t)|}\sum_{m=1}^{\mathcal{M}_i^n(t)} g_{m,(n-1)s+i}(\boldsymbol{\theta}_t)$, which is estimated as

$$\hat{g}^{\mathrm{e}}_{(n-1)s+i}(\boldsymbol{\theta}_t) = \frac{y_i^n(t)}{\lambda(t)\,|\mathcal{M}_i^n(t)|}, \quad \text{for } i \in [s],\, n \in [N]. \quad (7)$$

Estimated vector $\hat{\boldsymbol{g}}^{\mathrm{e}}(\boldsymbol{\theta}_t) \triangleq [\hat{g}^{\mathrm{e}}_1(\boldsymbol{\theta}_t) \cdots \hat{g}^{\mathrm{e}}_d(\boldsymbol{\theta}_t)]^T$ is then used to update the parameter vector as $\boldsymbol{\theta}_{t+1} = \boldsymbol{\theta}_t - \eta_t \hat{\boldsymbol{g}}^{\mathrm{e}}(\boldsymbol{\theta}_t)$.

Remark 1. *We remark here that the scheme in [10] imposes a stricter constraint $\bar{P}$ per iteration of the DSGD, i.e., at worker m we should have*

$$\frac{1}{N}\sum_{n=1}^{N} \mathbb{E}\left[||\boldsymbol{x}_m^n(t)||_2^2\right] \le \bar{P}, \quad \forall m \in [M], \forall t. \quad (8)$$

For fairness we generalize the scheme in [10] by defining the average power constraint per worker over iterations as in (3).

Algorithm 1 CWSA-DSGD

```
 1: Initialize $\boldsymbol{\theta}_1 = 0$ and $\boldsymbol{\Delta}_1(0) = \cdots = \boldsymbol{\Delta}_M(0) = 0$
 2: for $t = 1, \dots, T$ do
    • Workers do:
 3:     for $m = 1, \dots, M$ in parallel do
 4:         Compute $\boldsymbol{g}_m(\boldsymbol{\theta}_t)$ with respect to $\boldsymbol{u}_i \in \mathcal{B}_m$
 5:         $\boldsymbol{g}_m^{ec}(\boldsymbol{\theta}_t) = \boldsymbol{g}_m(\boldsymbol{\theta}_t) + \boldsymbol{\Delta}_m(t-1)$
 6:         $\boldsymbol{g}_m^{sp}(\boldsymbol{\theta}_t) = \mathrm{sparse}_k(\boldsymbol{g}_m^{ec}(\boldsymbol{\theta}_t))$
 7:         $\boldsymbol{\Delta}_m(t) = \begin{cases} \boldsymbol{g}_m^{ec}(\boldsymbol{\theta}_t) - \boldsymbol{g}_m^{sp}(\boldsymbol{\theta}_t), & \text{if } m \in \mathcal{M}(t), \\ \boldsymbol{g}_m(\boldsymbol{\theta}_t), & \text{if } m \notin \mathcal{M}(t) \end{cases}$
 8:         $\tilde{\boldsymbol{g}}_m(\boldsymbol{\theta}_t) = \boldsymbol{A}\boldsymbol{g}_m^{sp}(\boldsymbol{\theta}_t)$
 9:         $\boldsymbol{x}_m(t) = \boldsymbol{\alpha}_m(t) \cdot \tilde{\boldsymbol{g}}_m(\boldsymbol{\theta}_t)$
10:     end for
    • PS does:
11:     if $|\mathcal{M}(t)| \neq 0$ then
12:         $\hat{\boldsymbol{g}}^{\mathrm{w}}(\boldsymbol{\theta}_t) = \mathrm{AMP}_{\boldsymbol{A}}\left(\frac{\boldsymbol{y}(t)}{\gamma(t)|\mathcal{M}(t)|}\right)$
13:         $\boldsymbol{\theta}_{t+1} = \boldsymbol{\theta}_t - \eta_t \hat{\boldsymbol{g}}^{\mathrm{w}}(\boldsymbol{\theta}_t)$
14:     else
15:         $\boldsymbol{\theta}_{t+1} = \boldsymbol{\theta}_t$
16:     end if
17: end for
```

B. CWSA-DSGD

As opposed to ESA-DSGD, which aims to transmit all the gradient entries to the PS at each iteration, i.e., $N = \lceil d/s \rceil$, the CWSA-DSGD scheme proposed here applies gradient sparsification with error accumulation followed by a linear transformation to compress the gradients at each worker. The CWSA-DSGD scheme is designed for $N = 1$, i.e., the parameter vector is updated after each time slot. For ease of presentation, we drop the time slot parameter n. Algorithm 1 presents the CWSA-DSGD scheme.

At each iteration the workers sparsify their gradient estimates as described below. In order to retain the accuracy of their local gradient estimates, workers employ *error accumulation* [2], where the accumulated error vector at worker m until iteration t is denoted by $\boldsymbol{\Delta}_m(t-1) \in \mathbb{C}^d$, where we set $\boldsymbol{\Delta}_m(0) = \boldsymbol{0}$, $\forall m \in [M]$. After computing $\boldsymbol{g}_m(\boldsymbol{\theta}_t)$, worker m updates its estimate with the accumulated error as $\boldsymbol{g}_m^{ec}(\boldsymbol{\theta}_t) \triangleq \boldsymbol{g}_m(\boldsymbol{\theta}_t) + \boldsymbol{\Delta}_m(t-1)$, $m \in [M]$. Next, the workers apply gradient sparsification: worker m sets all but k elements with the highest magnitudes of vector $\boldsymbol{g}_m^{ec}(\boldsymbol{\theta}_t)$ to zero, and obtains a sparse vector $\boldsymbol{g}_m^{sp}(\boldsymbol{\theta}_t)$, $m \in [M]$. This k-level sparsification is represented by sparse_k in Algorithm 1, i.e., $\boldsymbol{g}_m^{sp}(\boldsymbol{\theta}_t) = \mathrm{sparse}_k(\boldsymbol{g}_m^{ec}(\boldsymbol{\theta}_t))$. Worker m, $m \in [M]$, then updates $\boldsymbol{\Delta}_m(t)$ as follows: $\boldsymbol{\Delta}_m(t) = \boldsymbol{g}_m^{ec}(\boldsymbol{\theta}_t) - \boldsymbol{g}_m^{sp}(\boldsymbol{\theta}_t)$ if worker m is scheduled, and $\boldsymbol{\Delta}_m(t) = \boldsymbol{g}_m(\boldsymbol{\theta}_t)$ if worker m is not scheduled (we will describe the worker scheduling scheme of CWSA-DSGD later). To transmit the sparse vectors over the limited-bandwidth channel, workers employ a random projection matrix, similarly to compressive sensing.

Assuming identically distributed datasets across the workers, the local gradient estimates will also follow identical distributions, and hence, will have similar sparsity patterns. A pseudo-random matrix $\boldsymbol{A} \in \mathbb{R}^{s \times d}$, with each entry i.i.d. according to $\mathcal{N}(0, 1/s)$, is generated and shared between the PS and the workers. At iteration t, worker m computes $\tilde{\boldsymbol{g}}_m(\boldsymbol{\theta}_t) \triangleq \boldsymbol{A}\boldsymbol{g}_m^{sp}(\boldsymbol{\theta}_t) \in \mathbb{C}^s$, and sends $\boldsymbol{x}_m(t) = \boldsymbol{\alpha}_m(t) \cdot \tilde{\boldsymbol{g}}_m(\boldsymbol{\theta}_t)$, where $\boldsymbol{\alpha}_m(t) \in \mathbb{C}^s$ is the power allocation vector, which is set to satisfy the average power constraint, $m \in [M]$. The i-th entry of power allocation vector $\boldsymbol{\alpha}_m(t)$ is set as follows:

$$\alpha_{m,i}(t) = \begin{cases} \frac{\gamma(t)}{h_{m,i}(t)}, & \text{if } \min\limits_{i \in [s]}\{|h_{m,i}(t)|^2\} \ge h^{\mathrm{w}}_{\mathrm{th}}(t), \\ 0, & \text{otherwise}, \end{cases} \quad (9)$$

for some $\gamma(t) \in \mathbb{R}$, where $h^{\mathrm{w}}_{\mathrm{th}}(t)$ is chosen to satisfy the average power constraint. According to (9), at each iteration t, only the workers whose channel gains are over a fixed threshold participate in the learning task. The selected set of workers for transmission at the t-th iteration is given by

$$\mathcal{M}(t) = \left\{ m \in [M] : \min_{i \in [s]}\{|h_{m,i}(t)|^2\} \ge h^{\mathrm{w}}_{\mathrm{th}}(t) \right\}. \quad (10)$$

Worker m is declared as *scheduled* at iteration t if $m \in \mathcal{M}(t)$. By substituting $\boldsymbol{x}_m(\boldsymbol{\theta}_t)$ and $\boldsymbol{\alpha}_m(t)$ into (2), it follows that

$$\boldsymbol{y}(t) = \gamma(t)\boldsymbol{A}\sum_{m \in \mathcal{M}(t)} \boldsymbol{g}_m^{sp}(\boldsymbol{\theta}_t) + \boldsymbol{z}(t). \quad (11)$$

The PS wants to recover $\frac{1}{|\mathcal{M}(t)|}\sum_{m=1}^{\mathcal{M}(t)} \boldsymbol{g}_m^{sp}(\boldsymbol{\theta}_t)$ from its noisy observations in (11). For this, using its knowledge of matrix $\boldsymbol{A}$ and the CSI, PS employs the approximate message passing (AMP) algorithm [13]. The AMP algorithm is represented by the $\mathrm{AMP}_{\boldsymbol{A}}$ in Algorithm 1. If $|\mathcal{M}(t)| \neq 0$, the estimate

$$\hat{\boldsymbol{g}}^{\mathrm{w}}(\boldsymbol{\theta}_t) = \mathrm{AMP}_{\boldsymbol{A}}\left(\frac{\boldsymbol{y}(t)}{\gamma(t)\,|\mathcal{M}(t)|}\right) \quad (12)$$

is used to update the parameter vector as $\boldsymbol{\theta}_{t+1} = \boldsymbol{\theta}_t - \eta_t \hat{\boldsymbol{g}}^{\mathrm{w}}(\boldsymbol{\theta}_t)$. On the other hand, if $|\mathcal{M}(t)| = 0$, the previous parameter vector is simply used as the new one, i.e., $\boldsymbol{\theta}_{t+1} = \boldsymbol{\theta}_t$.

Remark 2. *With ESA-DSGD, each worker transmits only the entries of its estimated gradient whose corresponding channel conditions are sufficiently good. Thus, the gradient vector is* *inherently sparsified, but only based on the channel gains, regardless of the importance of the gradient entries. Then the entire gradient vector is sent over the bandwidth-limited wireless MAC over orthogonal time periods. On the other hand, with CWSA-DSGD, each worker sends only $k \leq s$ important gradient entries, where the magnitude of each entry is regarded as the importance metric, by projecting the sparse gradient vector to a low-dimensional vector of length s. A worker is scheduled if all its channel gains are sufficiently good. We further highlight the error accumulation technique incorporated into CWSA-DSGD, whereas with ESA-DSGD, entries of the gradient vectors that are not sent are forgotten.*

C. Average Transmit Power Analysis

The average transmit power at worker m of the ESA-DSGD scheme in time slot n of iteration t based on (4), is given by

$$\begin{aligned}\bar{P}^{\mathrm{e},n}_{m}(t) &\triangleq \mathbb{E}\left[||\boldsymbol{x}^n_m(t)||_2^2\right] \\ &= \sum\nolimits_{i=1}^{s} \mathbb{E}\left[\left|\beta^n_{m,i}(t)\right|^2 \left|g_{m,(n-1)s+i}(\boldsymbol{\theta}_t)\right|^2\right]. \quad (13)\end{aligned}$$

We highlight that the entries of the gradient vector $\boldsymbol{g}^n_m(\boldsymbol{\theta}_t)$ do not depend on the channel gains $h^n_{m,i}(t)$, $\forall i, n, m$. It follows that, for $n \in [N]$, $m \in [M]$,

$$\bar{P}^{\mathrm{e},n}_{m}(t) = \sum\nolimits_{i=1}^{s} \left|g_{m,(n-1)s+i}(\boldsymbol{\theta}_t)\right|^2 \mathbb{E}\left[\left|\beta^n_{m,i}(t)\right|^2\right]. \quad (14)$$

Note that $\left|h^n_{m,i}(t)\right|^2$ follows an exponential distribution with mean σ_m^2, $\forall i, n, m$. Thus, we have

$$\mathbb{E}\left[\left|\beta^n_{m,i}(t)\right|^2\right] = \frac{\lambda(t)^2}{\sigma_m^2}\mathrm{E}_1(h^{\mathrm{e}}_{\mathrm{th}}(t)), \quad (15)$$

where $\mathrm{E}_1(x) \triangleq \int_x^\infty \frac{e^{-\tau}}{\tau} d\tau$. It follows that, $m \in [M], n \in [N]$,

$$\bar{P}^{\mathrm{e},n}_{m}(t) = \frac{\lambda^2(t)}{\sigma_m^2}\mathrm{E}_1(h^{\mathrm{e}}_{\mathrm{th}}(t)) P^n_{g,m}(t), \quad (16)$$

where we defined $P^n_{g,m}(t) \triangleq \|\boldsymbol{g}^n_m(\boldsymbol{\theta}_t)\|_2^2$. To satisfy the average power constraint, we set $\left\{\bar{P}^{\mathrm{e},n}_m(t), \forall m, n, t\right\}$, such that

$$\max_{m\in[M]} \left\{\frac{1}{NT}\sum\nolimits_{t=1}^{T}\sum\nolimits_{n=1}^{N} \bar{P}^{\mathrm{e},n}_{m}(t)\right\} \leq \bar{P}. \quad (17)$$

We follow a similar procedure to obtain the average transmit power of each worker of the proposed CWSA-DSGD scheme:

$$\begin{aligned}\bar{P}^{\mathrm{w}}_m(t) &\triangleq \mathbb{E}\left[||\boldsymbol{x}_m(t)||_2^2\right] = \sum\nolimits_{i=1}^{s} \mathbb{E}\left[|\alpha_{m,i}(t)|^2 \left|\tilde{g}_{m,i}(\boldsymbol{\theta}_t)\right|^2\right] \\ &= \sum\nolimits_{i=1}^{s} |\tilde{g}_{m,i}(\boldsymbol{\theta}_t)|^2 \mathbb{E}\left[|\alpha_{m,i}(t)|^2\right], \quad (18)\end{aligned}$$

where $\tilde{g}_{m,i}(\boldsymbol{\theta}_t)$ is the i-th entry of vector $\tilde{\boldsymbol{g}}_m(\boldsymbol{\theta}_t)$. The power allocation $\boldsymbol{\alpha}_m(t)$, given in (9), yields

$$\mathbb{E}\left[|\alpha_{m,i}(t)|^2\right] = \frac{\gamma^2(t)}{\sigma_m^2}\mathrm{E}_1(h^{\mathrm{w}}_{\mathrm{th}}(t)) e^{-(s-1)h^{\mathrm{w}}_{\mathrm{th}}(t)/\sigma_m^2}. \quad (19)$$

Thus, we have, for $m \in [M]$,

$$\bar{P}^{\mathrm{w}}_m(t) = \frac{\gamma^2(t)}{\sigma_m^2}\mathrm{E}_1(h^{\mathrm{w}}_{\mathrm{th}}(t)) e^{-(s-1)h^{\mathrm{w}}_{\mathrm{th}}(t)/\sigma_m^2} P_{\tilde{g},m}(t), \quad (20)$$

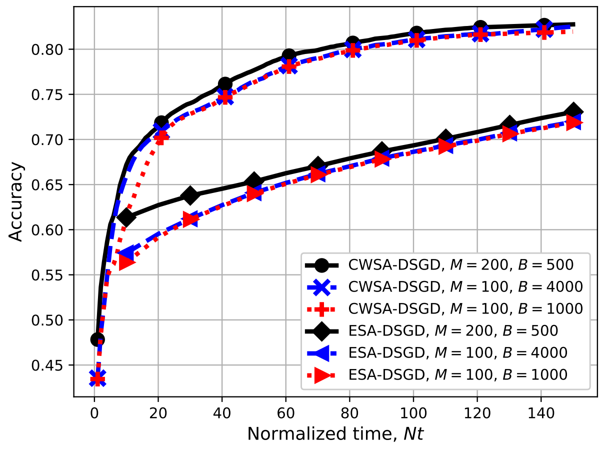

Fig. 1: Training accuracy of the ESA-DSGD and CWSA-DSGD algorithms for different M and B values.

TABLE I: Final test accuracy for schemes in Fig. 1.

ESA-DSGD, $(M, B) = (100, 1000)$	0.717
ESA-DSGD, $(M, B) = (100, 4000)$	0.718
ESA-DSGD, $(M, B) = (200, 500)$	0.729
CWSA-DSGD, $(M, B) = (100, 1000)$	0.818
CWSA-DSGD, $(M, B) = (100, 4000)$	0.82
CWSA-DSGD, $(M, B) = (200, 500)$	0.827

where $P_{\tilde{g},m}(t) \triangleq \|\tilde{\boldsymbol{g}}_m(\boldsymbol{\theta}_t)\|_2^2$. We set $\left\{\bar{P}^{\mathrm{w}}_m(t), \forall m, t\right\}$ such that $\max_{m\in[M]} \left\{\frac{1}{T}\sum_{t=1}^{T} \bar{P}^{\mathrm{w}}_m(t)\right\} \leq \bar{P}$.

IV. Experiments

Here we compare the performances of the ESA-DSGD and CWSA-DSGD schemes for the task of image classification. We run experiments on MNIST dataset [14] with $D = 60000$ training and 10000 test samples, and train a single layer neural network with $d = 7850$ parameters utilizing ADAM optimizer [15]. Since the data consists of real numbers, we treat the real and imaginary components of the channel as two independent parallel channels. A random set of B training data samples is assigned to each worker at the beginning of training. We consider $\sigma_m^2 = 1, \forall m \in [M]$, and for any number of channel uses s, we set $k = \lfloor s/2 \rfloor$. We note that the t-th iteration of the CWSA-DSGD scheme is equivalent to the $(t - N_s \lfloor (t-1)/N_s \rfloor)$-th time slot of the $(\lfloor (t-1)/N_s \rfloor + 1)$-th iteration of the ESA-DSGD scheme, where $N_s \triangleq \lceil d/s \rceil$. Thus, for fairness, we consider, $m \in [M]$,

$$\begin{aligned}&\max\nolimits_{m\in[M]}\{\bar{P}^{\mathrm{w}}_m(t)\} = \\ &\max\nolimits_{m\in[M]}\left\{\bar{P}^{\mathrm{e},t-N_s\lfloor (t-1)/N_s\rfloor}_m(\lfloor (t-1)/N_s \rfloor + 1)\right\}, \quad (21)\end{aligned}$$

where $\bar{P}^{\mathrm{e},n}_m(t)$ and $\bar{P}^{\mathrm{w}}_m(t)$ are given in (16) and (20), respectively. We also consider $\lambda(\lfloor (t-1)/N_s \rfloor + 1) = 1$, and we find $\gamma(t)$ according to (21), $\forall t$. The performance is measured as the accuracy with respect to the training dataset versus normalized time Nt, and the final accuracy with respect to

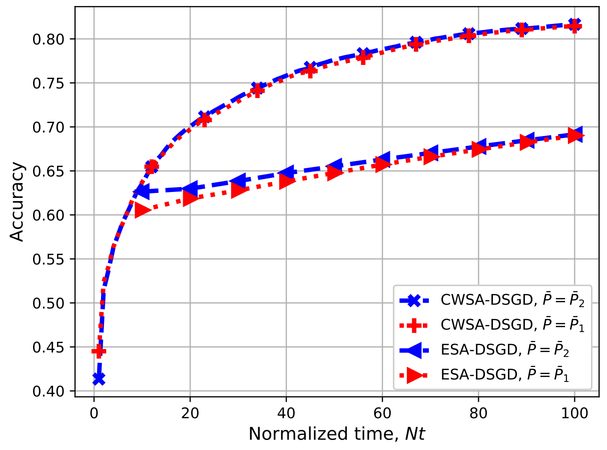

Fig. 2: Training accuracy of the ESA-DSGD and CWSA-DSGD algorithms for different $\bar{P}$ values.

TABLE II: Final test accuracy for schemes in Fig. 2.

ESA-DSGD, $\bar{P} = \bar{P}_1$	0.689
ESA-DSGD, $\bar{P} = \bar{P}_2$	0.691
CWSA-DSGD, $\bar{P} = \bar{P}_1$	0.814
CWSA-DSGD, $\bar{P} = \bar{P}_2$	0.815

the test samples, i.e., test accuracy, based on the parameter vector obtained after a total of NT time slots.

In Fig. 1, we compare the performances of ESA-DSGD and CWSA-DSGD algorithms for different (M, B) pairs, $(M, B) \in \{(100, 1000), (100, 4000), (200, 500)\}$. We consider $s = d/10$, which results in $N = 10$ for ESA-DSGD, and we set $h^{\mathrm{w}}_{\mathrm{th}}(t) = 10^{-3}$, and $h^{\mathrm{e}}_{\mathrm{th}}(t) = 3.2 \times 10^{-2}$, $\forall t$. We consider $NT = 150$ time slots in total, and the resultant average transmit power for both ESA-DSGD and CWSA-DSGD schemes is $\bar{P} = 0.36$. We present the final test accuracies for the settings under consideration in Table I. As it can be seen, CWSA-DSGD performs significantly better than ESA-DSGD. The main reasons for the degradation of the ESA-DSGD over CWSA-DSGD are i) scheduling gradient entries for transmission only based on the channel gains; ii) sending the entire gradient vectors of relatively huge dimensions; iii) ignoring the gradient entries which have not been sent due to bad channel conditions. By comparing the cases $(M, B) = (100, 1000)$ and $(M, B) = (200, 500)$, we observe that the performance of both schemes improve by increasing M while keeping the total size of the dataset, MB, constant. This is because increasing M provides additional power introduced by each worker and increases the robustness of the estimation against noise. For $M = 100$, the improvement of both schemes is negligible by increasing B from $B = 1000$ to $B = 4000$, and both schemes perform better with $(M, B) = (200, 500)$ compared to $(M, B) = (100, 4000)$, despite 4 times reduction in MB.

In Fig. 2, we compare ESA-DSGD and CWSA-DSGD algorithms for different average transmit power values $\bar{P}_1 = 0.36$ and $\bar{P}_2 = 0.59$. Values $\bar{P} = \bar{P}_1$ and $\bar{P} = \bar{P}_2$ are the results of setting $h^{\mathrm{w}}_{\mathrm{th}}(t) = 1.25 \times 10^{-3}$ and $h^{\mathrm{e}}_{\mathrm{th}}(t) = 3.2 \times 10^{-2}$, $\forall t$, and $h^{\mathrm{w}}_{\mathrm{th}}(t) = 5 \times 10^{-4}$ and $h^{\mathrm{e}}_{\mathrm{th}}(t) = 4.9 \times 10^{-3}$, $\forall t$, respectively. We consider $s = d/10$, $M = 200$, $B = 500$, and $NT = 100$. The final test accuracies of different schemes under consideration are presented in Table II. We highlight the significant superiority of CWSA-DSGD over ESA-DSGD. Observe that reducing the power slightly degrades the performance; however, the degradation of CWSA-DSGD with power is marginal, due to a more efficient use of the available power to transmit only the more important elements of the gradient vectors by the workers.

V. Conclusions

We have studied distributed machine learning at the wireless edge, where M workers with limited datasets communicate with the PS over a fading MAC to minimize a loss function by performing DSGD. The PS updates the parameter vector, and shares it with the workers. We considered an analog transmission approach from the workers to the PS. We have proposed gradient sparsification with error accumulation followed by compressive sensing to reduce the typically very large parameter vector dimension to the limited channel bandwidth at each worker. We have designed a power allocation scheme to align the received vectors at the PS while satisfying the average power constraints. This analog approach allows a much more efficient use of the limited channel bandwidth, and benefits from the “beamforming effect” of superposed signals. Numerical results have shown significant improvements in the performance compared to the state-of-the-art.

References

[1] D. Alistarh, D. Grubic, J. Z. Li, R. Tomioka, and M. Vojnovic, “QSGD: Communication-efficient SGD via randomized quantization and encoding,” in *NIPS*, Long Beach, CA, Dec. 2017, pp. 1709–1720.
[2] F. Seide, H. Fu, J. Droppo, G. Li, and D. Yu, “1-bit stochastic gradient descent and its application to data-parallel distributed training of speech DNNs,” in *INTERSPEECH*, Singapore, Sep. 2014, pp. 1058–1062.
[3] S. Gupta, A. Agrawal, K. Gopalakrishnan, and P. Narayanan, “Deep learning with limited numerical precision,” in *ICML*, Jul. 2015.
[4] N. Strom, “Scalable distributed DNN training using commodity gpu cloud computing,” in *INTERSPEECH*, 2015.
[5] M. M. Amiri and D. Gündüz, “Computation scheduling for distributed machine learning with straggling workers,” *arXiv:1810.09992 [cs.DC]*.
[6] A. F. Aji and K. Heafield, “Sparse communication for distributed gradient descent,” *arXiv:1704.05021v2 [cs.CL]*, Jul. 2017.
[7] F. Sattler et al., “Sparse binary compression: Towards distributed deep learning with minimal communication,” *arXiv:1805.08768v1 [cs.LG]*.
[8] H. B. McMahan et al., “Communication-efficient learning of deep networks from decentralized data,” in *Proc. AISTATS*, 2017.
[9] T. Chen et al., “LAG: Lazily aggregated gradient for communication-efficient distributed learning,” *arXiv:1805.09965 [stat.ML]*, May 2018.
[10] G. Zhu, Y. Wang, and K. Huang, “Low-latency broadband analog aggregation for federated edge learning,” *arXiv:1812.11494 [cs.IT]*.
[11] K. Yang, T. Jiang, Y. Shi, and Z. Ding, “Federated learning via over-the-air computation,” *arXiv:1812.11750 [cs.LG]*, Jan. 2019.
[12] M. M. Amiri and D. Gündüz, “Machine learning at the wireless edge: Distributed stochastic gradient descent over-the-air,” *arXiv:1901.00844 [cs.DC]*, Jan. 2019.
[13] D. L. Donoho, A. Maleki, and A. Montanari, “Message-passing algorithms for compressed sensing,” *Proc. Nat. Acad. Sci. USA*, vol. 106, no. 45, pp. 18 914–18 919, Nov. 2009.
[14] Y. LeCun, C. Cortes, and C. Burges, “The MNIST database of handwritten digits,” *http://yann.lecun.com/exdb/mnist/*, 1998.
[15] D. P. Kingma and J. Ba, “Adam: A method for stochastic optimization,” *arXiv:1412.6980v9 [cs.LG]*, Jan. 2017.