RESOURCE-EFFICIENT FEDERATED LEARNING ROBUST TO COMMUNICATION ERRORS

Ehsan Lari⋆*, Vinay Chakravarthi Gogineni*⋆*, Reza Arablouei*† *and Stefan Werner*⋆

⋆Department of Electronic System, Norwegian University of Science and Technology, Trondheim, Norway
†Data61, CSIRO, Pullenvale QLD 4069, Australia

ABSTRACT

The effectiveness of federated learning (FL) in leveraging distributed datasets is highly contingent upon the accuracy of model exchanges between clients and servers. Communication errors caused by noisy links can negatively impact learning accuracy. To address this issue, we present an FL algorithm that is robust to communication errors while reducing the communication load on clients. To derive the proposed algorithm, we consider a weighted least-squares regression problem as a motivating example. We cast the considered problem as a distributed optimization problem over a federated network, which employs random scheduling to enhance communication efficiency, and solve it using the alternating direction method of multipliers. To improve robustness, we eliminate the local dual parameters and reduce the number of global model exchanges via a change of variable. We analyze the mean convergence of our proposed algorithm and demonstrate its effectiveness compared with related existing algorithms via simulations.

1. INTRODUCTION

With the increasing prevalence of smart devices, big data is becoming more ubiquitous. Learning from big data can enhance the decision-making capability of the end-users [1, 2]. However, this is challenging as the data is stored locally on edge devices and moving it to the cloud or a central server may raise privacy/security or excessive resource utilization concerns. Federated learning (FL) is a distributed learning paradigm that allows edge devices to collaboratively learn a shared global model using their locally-stored data without compromising their data privacy [3, 4]. FL is increasingly popular due to its ability to handle heterogeneous data and devices [5]. Data heterogeneity may refer to data being non-IID or imbalance in client data used to learn the global model [6, 7]. Device heterogeneity relates to diversity in storage, energy, computational, or communication resources of the clients participating in FL [8, 9].

The federated average (FedAvg) algorithm [3] is a popular baseline FL method. In FedAvg, the global iteration round begins with the server sharing its aggregated model with a subset of all clients selected uniformly at random, typically over a wireless network. After receiving the aggregated model from the server, the clients perform local learning to update the model and share the updated model with the server. Finally, the server receives the local models and aggregates them to produce a new global model. This process repeats until a specific convergence criterion is met. Many FL methods, including FedAvg, have been studied in the literature considering different aspects such as data privacy [10], model poisoning attacks [11], and communication efficiency [12]. However, most FL algorithms assume ideal communication links and do not take communication noise or error into account [13–17].

This work was supported by the Research Council of Norway.

When the communication channels between the server and the edge devices are noisy, the server receives noisy local updates due to uplink noise, and each client receives a noisy version of the aggregated model from the server due to downlink noise [18–20]. Using models that are corrupted by communication noise/error can deteriorate the quality of the learned model. Many works on FL have primarily focused on the uplink noise [21, 22]. In [23], the impact of downlink noise on FL is investigated. These studies show that the performance of gradient-descent-based FL algorithms can degrade when noise is present in the communication channels. In [19], a new loss function is proposed for FL using the first-order derivative of the loss function as a regularizer to overcome the problem of additive noise in communication links. In [23], two approaches are proposed to make FL more robust to the downlink noise. The first approach is based on using a quantization technique alongside transmitting the global model update via digital links and employing channel coding with a common rate. The other approach is based on an analog downlink transmission scheme where the server transmits an uncoded global model update.

The authors of [24, 25] propose that by controlling the scale of the communication signal-to-noise ratio, the noise can be tolerated and the convergence rate of FedAvg with perfect communication links can be maintained. However, they do not consider any countermeasure for the effects of the noise. In [26], the authors use precoding and scaling upon transmissions to alleviate the ill effects of noisy channels and ensure the convergence of their algorithm. These and some other similar methods proposed to deal with link noise in FL usually require additional resources on the client side. This can be counterproductive as, in FL, clients often operate with limited resources in terms of power/energy, memory, or computational capacity.

FL approaches based on the alternating direction method of multipliers (ADMM) can exhibit some robustness to additive communication noise due to the nature of their design [27]. However, they require all clients to participate in every FL round, which may be impractical in real-life scenarios when the clients are resource-constrained or heterogeneous edge devices. Therefore, there is a great demand for FL algorithms that are robust to noise in communication links with minimal extra communication or computational requirements.

In this paper, we propose a resource-efficient ADMM-based FL algorithm that is robust to communication noise/error while imposing no additional computational burden on the participating clients. We consider the presence of noise in both uplink and downlink communications. Considering the weighted least-squares (WLS) regression problem as a motivating example, we develop our proposed FL algorithm by iteratively solving an appropriately-formulated distributed convex optimization problem via the ADMM. To achieve communication efficiency, we employ random scheduling of the clients. Furthermore, to prevent error accumulation from degrading the learning, we communicate a linear combination of the last two global model updates as well as eliminating the dual model parameters at all participating edge devices. Through theoretical analysis, we show that the convergence of the proposed algorithm is ensured when the server chooses a

random subset of the clients at each iteration even with noisy communication links. Our simulation results also attest to the effectiveness of the proposed algorithm in comparison with the state of the art as well as corroborating our theoretical findings.

2. PROBLEM FORMULATION

We consider a federated network consisting of N edge devices (clients), each directly connected to a server. Each client i has a dataset denoted by $\mathcal{D}_i = \{\mathbf{H}_i, \mathbf{y}_i\}$ where $\mathbf{y}_i$ is a column vector with d_i entries and $\mathbf{H}_i$ is a matrix of size $d_i \times L$. For every client i, a linear regression model relating $\mathbf{H}_i$ to $\mathbf{y}_i$ can be described as

$$\mathbf{y}_i = \mathbf{H}_i\mathbf{x} + \boldsymbol{\nu}_i, \tag{1}$$

where $\mathbf{x}$ is the global regression parameter vector of size $L \times 1$ and $\boldsymbol{\nu}_i$ represents perturbation/noise.

The main goal of FL is to estimate the parameter vector $\mathbf{x}$ by collaboratively minimizing a global objective function over the federated network. To this end, we define a global WLS estimation problem in the federated setting as

$$\min_{\{\mathbf{x}_i\}} \sum_{i=1}^{N} \mathcal{J}_i(\mathbf{x}_i) \;\; \text{s.t. } \mathbf{x}_i = \mathbf{x},\; i = 1, 2, \cdots, N, \tag{2}$$

where $\mathcal{J}_i(\mathbf{x}) = \|\mathbf{y}_i - \mathbf{H}_i\mathbf{x}\|^2_{\mathbf{W}_i}$ is the local objective function for estimating $\mathbf{x}$ at client i and $\mathbf{W}_i$ is the error weight matrix of client i. In addition, $\mathbf{x}_i$ represents the local model estimate at client i and $\mathbf{x}$ denotes the global model estimate.

We use the ADMM to solve (2) whose associated augmented Lagrangian function can be expressed as

$$\sum_{i=1}^{N} \mathcal{L}_i(\mathbf{x}, \mathbf{x}_i, \boldsymbol{\pi}_i) = \sum_{i=1}^{N} \mathcal{J}_i(\mathbf{x_i}) + \langle \mathbf{x}_i - \mathbf{x}, \boldsymbol{\pi}_i \rangle + \frac{\rho_i}{2}\|\mathbf{x}_i - \mathbf{x}\|_2^2, \tag{3}$$

where $\boldsymbol{\pi}_i$ and $\rho_i > 0$ are the Lagrange multiplier vector and the penalty parameter associated with client i, respectively. Therefore, we obtain the following recursions at each client

$$\boldsymbol{\pi}_{i,k} = \boldsymbol{\pi}_{i,k-1} + \rho_i(\mathbf{x}_{i,k} - \mathbf{x}_k) \tag{4a}$$

$$\mathbf{x}_{i,k+1} = \hat{\mathbf{x}}_i - \mathbf{N}_i^{-1}(\boldsymbol{\pi}_{i,k} - \rho_i\mathbf{x}_k) \tag{4b}$$

and at the server

$$\mathbf{x}_{k+1} = \frac{1}{\sum_{i=1}^{N} \rho_i} \sum_{i=1}^{N} (\rho_i\mathbf{x}_{i,k+1} + \boldsymbol{\pi}_{i,k}) \tag{5}$$

where we define $\mathbf{N}_i = 2\mathbf{H}_i^\intercal\mathbf{W}_i\mathbf{H}_i + \rho_i\mathbf{I}$ and $\hat{\mathbf{x}}_i = 2\mathbf{N}_i^{-1}\mathbf{H}_i^\intercal\mathbf{W}_i\mathbf{y}_i$, and the index k denotes the iteration number. In the above algorithm, after performing local learning, i.e., (4a) and (4b), each client shares the local estimate $\rho_i\mathbf{x}_{i,k+1} + \boldsymbol{\pi}_{i,k}$ with the server. The server then obtains the global estimate as in (5) and broadcasts it to every client while the FL process continues.

In the formulation (4) and (5), there is a need to send both primal and dual model updates to the server in order for the server to be able to aggregate the global model update. However, the information in the dual update can be incorporated inside the primal update using a careful choice of the initial value. Therefore, we can reformulate the recursions (4)-(5) as

$$\mathbf{x}_{i,k+1} = (\mathbf{I} - \rho_i\mathbf{N}_i^{-1})\mathbf{x}_{i,k} + \rho_i\mathbf{N}_i^{-1}(2\mathbf{x}_k - \mathbf{x}_{k-1}) \tag{6a}$$

$$\mathbf{x}_{k+1} = \frac{1}{\sum_{i=1}^{N} \rho_i} \sum_{i=1}^{N} \rho_i\mathbf{x}_{i,k+1}. \tag{6b}$$

via initializing $\mathbf{x}_{-1} = \mathbf{0}$, $\boldsymbol{\pi}_{i,-1} = \mathbf{0}$, and $\mathbf{x}_{i,0} = \hat{\mathbf{x}}_i$ and eliminating the Lagrange multipliers $\boldsymbol{\pi}_{i,k}$. Defining $\mathbf{s}_{i,k} = 2\mathbf{x}_{i,k} - \mathbf{x}_{i,k-1}$ and $\mathbf{s}_k = 2\mathbf{x}_k - \mathbf{x}_{k-1}$, we can further rewrite (6) as

$$\mathbf{x}_{i,k+1} = (\mathbf{I} - \rho_i\mathbf{N}_i^{-1})\mathbf{x}_{i,k} + \rho_i\mathbf{N}_i^{-1}\mathbf{s}_k \tag{7a}$$

$$\mathbf{s}_{i,k+1} = 2\mathbf{x}_{i,k+1} - \mathbf{x}_{i,k} \tag{7b}$$

$$\mathbf{s}_{k+1} = \frac{1}{\sum_{i=1}^{N} \rho_i} \sum_{i=1}^{N} \rho_i\mathbf{s}_{i,k+1}. \tag{7c}$$

In this algorithm, the clients share $\mathbf{s}_{i,k+1}$ with the server, and the server broadcasts $\mathbf{s}_{k+1}$ to the clients. As we will show later, this formulation is favorable when the communication links are unideal.

The parameter exchanges between the clients and the server are often carried out over wireless communication channels. Therefore, both uplink and downlink channels may be corrupted by noise. In the downlink, the clients receive noisy versions of the aggregated model updates from the server, i.e., $\tilde{\mathbf{s}}_k^i = \mathbf{s}_k + \boldsymbol{\zeta}_k^i$ where $\boldsymbol{\zeta}_k^i$ denotes the downlink noise of client i at iteration k. In the uplink, the server receives a noisy version of the local model update of each client, i.e., $\tilde{\mathbf{s}}_{i,k} = \mathbf{s}_{i,k} + \boldsymbol{\eta}_{i,k}$ where $\boldsymbol{\eta}_{i,k}$ denotes the uplink noise of client i at iteration k.

Comparing the recursions (4)-(5) and (6) with (7), we hypothesize that introducing $\mathbf{s}_{i,k+1}$ as a linear combination of $\mathbf{x}_{i,k+1}$ and $\mathbf{x}_{i,k}$ and sending it instead of $\mathbf{x}_{i,k+1}$ can result in less noise corruption in the estimates of the clients due to using a single noisy global estimate rather that two. Hence, it can lead to improved performance in terms of robustness against additive communication noise.

However, the recursions (7) require all clients to take part in a global model update iteration. In FL, the clients may have different communication capabilities due to having limited resources. Therefore, the participation of all clients at each global update round may come at a considerable cost, e.g., slow convergence time or increased resource utilization. To tackle this, the server may implement a random scheduling of the clients and have only a subset of the clients denoted by $\mathcal{S}_k$ participate in model aggregation at each iteration k. The scheduling can lower the communication overhead of FL significantly. Due to the choosing of a random subset of the clients, some clients may not be selected at two consecutive iterations. As a result, the recursions (7) will fail to work as two consecutive updates $\mathbf{x}_{i,k+1}$ and $\mathbf{x}_{i,k}$ may not be available at the client and it is not always possible to calculate the signal $\mathbf{s}_{i,k+1}$ at the clients, i.e., if client i is selected at $k+1$ and $k' \neq k$, then $2\mathbf{x}_{i,k+1} - \mathbf{x}_{i,k'} \neq \mathbf{s}_{i,k+1}$. Therefore, we consider sending the local model updates $\mathbf{x}_{i,k+1}$ instead of $\mathbf{s}_{i,k+1}$ in order to guarantee convergence.

3. RESOURCE-EFFICIENT FEDERATED LEARNING OVER NOISY CHANNELS

Communication efficiency is essential for FL in real-world applications, as it directly affects its scalability and cost-effectiveness. The efficiency of communication is closely tied to the amount of data that needs to be transmitted among the clients and the server during the model training process. When communication load is high, it can lead to increased resource usage and longer training times, resulting in higher costs and decreased system utility. Therefore, minimizing data transmissions while maintaining high accuracy is a critical challenge in FL.

Random scheduling of the clients for communication is one way to improve the communication efficiency in FL. However, as it is managed by the server, at any given time, two consecutive updates may not be available at the client. Hence, it may not be possible to calculate $\mathbf{s}_{i,k+1}$ at all clients and iterations. Therefore, the recursions (7) may fail to converge. As a solution, we propose to calculate the local model

Algorithm 1 : `RERCE-Fed`. N clients, penalty parameters ρ_i, set of all clients $\mathcal{S}$.

Initialization: global model $\mathbf{x}_0 = \mathbf{x}_{-1} = \mathbf{0}$, local models $\mathbf{x}_{i,0} = \hat{\mathbf{x}}_i$

For $k = 1, ...$

The server randomly selects a subset $\mathcal{S}_k$ of its clients and sends the aggregated global model $\mathbf{s}_k = 2\mathbf{x}_k - \mathbf{x}_{k-1}$ to them.

Client Local Update:

Each client $i \in \mathcal{S}_k$ receives a noisy version of $\mathbf{s}_k$ and updates its model via (8a), then sends its updated model $\mathbf{x}_{i,k+1}$ to the server.

Aggregation at the Server:

The server receives noisy versions of the locally updated models and aggregates them via (8b).

EndFor

updates at the clients selected by random scheduling $i \in \mathcal{S}_k$ and send them to the server. The server then aggregates the received local updates and sends $\mathbf{s}_k = 2\mathbf{x}_k - \mathbf{x}_{k-1}$ to the new set of selected clients. Note that $\mathbf{s}_k$ is corrupted with different noise for two different sets of clients and the last two global iterations.

In each global iteration k, the selected clients receive $\tilde{\mathbf{s}}_k^i$ from the server and update their model by (8a). Then, the server receives the signal $\tilde{\mathbf{x}}_{i,k+1}$ from the selected clients and aggregates the received signal via (8b) and broadcasts the latest global update to the selected client in the next iteration. The clients that are not selected maintain their last update until they are selected again. Therefore, the recursions of the proposed resource-efficient FL algorithm robust to communication errors, called RERCE-Fed, are expressed as

$$\mathbf{x}_{i,k+1} = \mathbf{x}_{i,k} - a_{i,k}\rho_i \mathbf{N}_i^{-1}\mathbf{x}_{i,k} + a_{i,k}\rho_i \mathbf{N}_i^{-1}\tilde{\mathbf{s}}_k^i \tag{8a}$$

$$\mathbf{x}_{k+1} = \frac{1}{\sum\limits_{i=1}^{N} a_{i,k}\rho_i} \sum_{i=1}^{N} a_{i,k}\rho_i \tilde{\mathbf{x}}_{i,k+1}, \tag{8b}$$

where $a_{i,k}$ is the variable that represents random scheduling such that $a_{i,k} = 1$ when the client i in selected by the server in iteration k, i.e., $i \in \mathcal{S}_k$, and $a_{i,k} = 0$ otherwise. We summarize the proposed RERCE-Fed in Algorithm 1. In the following section, we study its mean convergence.

4. CONVERGENCE ANALYSIS

To facilitate the analysis, we define the extended optimal global model as $\mathbf{x}_e^\star = \mathbf{1}_{2N} \otimes \mathbf{x}^\star$ and the vector containing the client model estimate as $\tilde{\mathbf{x}}_{e,k} = \text{col}\{\tilde{\mathbf{x}}_{1,k}, \cdots, \tilde{\mathbf{x}}_{N,k}\}$, where $\mathbf{1}_{2N}$ is the $2N \times 1$ vector of all ones, $\mathbf{x}^\star$ is the optimal solution to (2), $\otimes$ is the Kronecker product, and $\text{col}\{\cdot\}$ denotes column-wise stacking.

With ideal communication links, it can be shown that the iterates $\mathbf{x}_{i,k}$ and $\mathbf{x}_k$ converge as $k \to \infty$. Our goal here is to show that they still converge when the communication channels are noisy.

Substituting (8b) in (8a), the global recursion of the proposed algorithm can be stated as

$$\begin{bmatrix} \mathbf{x}_{e,k+1} \\ \mathbf{x}_{e,k} \end{bmatrix} = \boldsymbol{\mathcal{A}}_k \begin{bmatrix} \mathbf{x}_{e,k} \\ \mathbf{x}_{e,k-1} \end{bmatrix} + \boldsymbol{\zeta}_k + \boldsymbol{\eta}_k, \tag{9}$$

where

$$\boldsymbol{\mathcal{A}}_k = \left[\begin{array}{c:c} \boldsymbol{\mathcal{A}}_{k,1} & \boldsymbol{\mathcal{A}}_{k,2} \\ \hdashline \boldsymbol{\mathcal{A}}_{k,3} & \boldsymbol{\mathcal{A}}_{k,4} \end{array}\right] \tag{10}$$

and the noise vectors $\boldsymbol{\zeta}_k$ and $\boldsymbol{\eta}_k$ stack

$$a_{i,k}\rho_i \mathbf{N}_i^{-1}\boldsymbol{\zeta}_{i,k} \tag{11}$$

and

$$a_{i,k}\rho_i \mathbf{N}_i^{-1} \sum_{j=1}^{N} \left(\frac{2a_{j,k-1}\rho_j \boldsymbol{\eta}_{j,k-1}}{\sum\limits_{u=1}^{N} a_{u,k-1}\rho_u} - \frac{a_{j,k-2}\rho_j \boldsymbol{\eta}_{j,k-2}}{\sum\limits_{u=1}^{N} a_{u,k-2}\rho_u} \right), \tag{12}$$

respectively, for $1 \le i \le N$ and zero for $N+1 \le i \le 2N$.

The value of $\boldsymbol{\mathcal{A}}_k \in \mathbb{R}^{2LN \times 2LN}$ depends on the iteration number k as the server selects a random number of clients at each iteration. Its sub-matrices of size $LN \times LN$ are block matrices whose $L \times L$ client-wise sub-matrices are

$$[\boldsymbol{\mathcal{A}}_{k,1}]_{ii} = \mathbf{I} - a_{i,k}\rho_i \mathbf{N}_i^{-1} + 2a_{i,k}a_{i,k-1} \frac{\rho_i^2 \mathbf{N}_i^{-1}}{\sum\limits_{u=1}^{N} a_{u,k-1}\rho_u}, \tag{13a}$$

$$[\boldsymbol{\mathcal{A}}_{k,1}]_{ij} = 2a_{i,k}a_{j,k-1} \frac{\rho_i\rho_j \mathbf{N}_i^{-1}}{\sum\limits_{u=1}^{N} a_{u,k-1}\rho_u}, \tag{13b}$$

$$[\boldsymbol{\mathcal{A}}_{k,2}]_{ij} = -a_{i,k}a_{j,k-2} \frac{\rho_i\rho_j \mathbf{N}_i^{-1}}{\sum\limits_{u=1}^{N} a_{u,k-2}\rho_u}, \tag{13c}$$

$\boldsymbol{\mathcal{A}}_{k,3} = \mathbf{I}$, and $\boldsymbol{\mathcal{A}}_{k,4} = \mathbf{0}$.

Applying the expectation operator $\mathbb{E}[\cdot]$ to both sides of (9) and considering the fact that the noises are zero-mean, we obtain

$$\mathbb{E}\begin{bmatrix} \mathbf{x}_{e,k+1} \\ \mathbf{x}_{e,k} \end{bmatrix} = \boldsymbol{\mathcal{A}}_k \mathbb{E}\begin{bmatrix} \mathbf{x}_{e,k} \\ \mathbf{x}_{e,k-1} \end{bmatrix}. \tag{14}$$

Since $\boldsymbol{\mathcal{A}}_k$ is a right-stochastic matrix as the entries of all its columns add up to one, (14) can be recursively unfolded as

$$\mathbb{E}\begin{bmatrix} \mathbf{x}_{e,k+1} \\ \mathbf{x}_{e,k} \end{bmatrix} = \boldsymbol{\mathcal{A}}'_k \mathbb{E}\begin{bmatrix} \mathbf{x}_{e,1} \\ \mathbf{x}_{e,0} \end{bmatrix},$$

where $\boldsymbol{\mathcal{A}}'_k = \prod_{i=1}^{k} \boldsymbol{\mathcal{A}}_i$. The matrix $\boldsymbol{\mathcal{A}}'_k$ is right-stochastic as the multiplication of right-stochastic matrices produces a right-stochastic matrix, i.e.,$\boldsymbol{\mathcal{A}}'_k\mathbf{1} = \left(\prod_{i=1}^{k} \boldsymbol{\mathcal{A}}_i\right)\mathbf{1} = \mathbf{1}$. An important property of a right-stochastic matrix is that all its eigenvalues λ_i satisfy $|\lambda_i| \le 1$. Therefore, $\boldsymbol{\mathcal{A}}'_k$ is stable and the recursions (9) converge.

Furthermore, defining $\mathbb{E}[\epsilon_{e,k+1}] = \mathbf{x}_e^\star - \mathbb{E}[\mathbf{x}_{e,k+1}^\intercal \mathbf{x}_{e,k}^\intercal]^\intercal$, and utilizing the fact that $\mathbf{x}_e^\star = \boldsymbol{\mathcal{A}}'_k\mathbf{x}_e^\star$, $\mathbb{E}[\epsilon_{e,k+1}]$ can be recursively expressed as $\mathbb{E}[\epsilon_{e,k+1}] = \boldsymbol{\mathcal{A}}'_k\mathbb{E}[\epsilon_{e,1}]$. As both $\boldsymbol{\mathcal{A}}'_k$ and $\mathbb{E}[\epsilon_{e,1}]$ are bounded, the expected error $\mathbb{E}[\epsilon_{e,k+1}]$ is bounded and the proposed RERCE-Fed algorithm converges. In the next section, we verify the convergence and robustness of the proposed algorithm to link noise via numerical simulations.

5. SIMULATION RESULTS

In this section, we conduct a series of experiments to examine the performance of the proposed RERCE-Fed. We consider a scenario having $N = 100$ clients connected to a server. The clients aim to learn a model $\mathbf{x}$ of dimension $L = 128$. To induce data imbalance, we draw the local dataset size of each client, d_i, randomly from a uniform distribution, i.e., $d_i \in \mathcal{U}(50, 90)$. Every client i has imbalanced synthetic non-IID data $\{\mathbf{H}_i, \mathbf{y}_i\}$ with the matrices $\mathbf{H}_i$ drawn from a normal distribution $\mathcal{N}(\mu_{\mathbf{H}_i}, \sigma^2_{\mathbf{H}_i})$ where $\mu_{\mathbf{H}_i} \in \mathcal{U}(-0.5, 0.5)$ and $\sigma^2_{\mathbf{H}_i} \in \mathcal{U}(0.5, 1.5)$. We set the weight matrices at each client i to the inverse covariance matrix of $\mathbf{y}_i$, i.e., $\mathbf{W}_i = \Sigma_{\mathbf{y}_i}^{-1} = \mathbb{E}[(\mathbf{y}_i - \mathbb{E}[\mathbf{y}_i])(\mathbf{y}_i - \mathbb{E}[\mathbf{y}_i])^\intercal]^{-1}$. The parameter vector $\mathbf{x}$ is drawn from a normal distribution $\mathcal{N}(0, 1)$. The observation noise $\boldsymbol{\nu}_i$ is taken as zero-mean IID Gaussian with variance

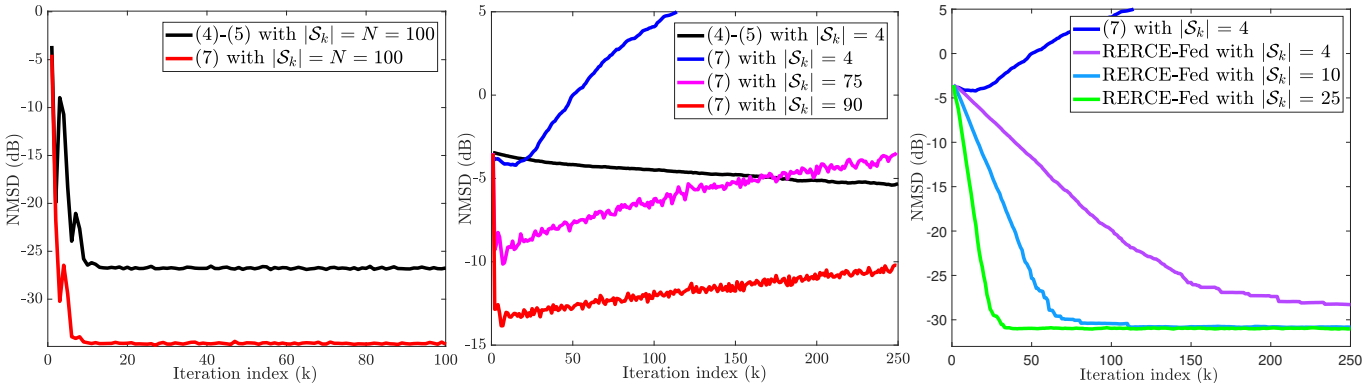

Fig. 1. Normalized mean-square deviation (NMSD) versus iteration index: (a) Learning curves of (4)-(5) and (7) for $|\mathcal{S}_k| = N = 100$. (b). Learning curves of (4)-(5) and (7) for different values of $|\mathcal{S}_k|$. (c). Learning curves of the proposed RERCE-Fed for different values of $|\mathcal{S}_k|$.

10^{-4} for each client. The additive noise in both uplink and downlink is zero-mean IID white Gaussian with variance 6.25×10^{-4}. In all simulated algorithms, the penalty parameter is set to $\rho_i = 1$ for all clients. The server randomly selected a subset of clients with uniform probability in every iteration k. We evaluate the performance of the considered algorithms via the network-wide average normalized mean-square deviation (NMSD) defined at iteration k as

$$\text{NMSD}(k) = \frac{1}{N}\sum_{i=1}^{N}\frac{\|\mathbf{x}_{i,k} - \mathbf{x}\|_2^2}{\|\mathbf{x}\|_2^2}. \tag{15}$$

The learning curves (i.e., NMSD in dB vs. iteration number k) presented in the following figures are obtained by averaging over 100 independent experiments.

In our first experiment, we simulate (4)-(5) and (7) solving the regression problem outlined in section 2 when all clients participate in FL, i.e., $|\mathcal{S}_k| = N = 100$. We present the corresponding learning curves in Fig. 1(a). We notice that (7) exhibits improved performance (7dB improvement) over (4)-(5) in the presence of noisy communication links.

In many practical applications, the FL clients operate under resource constraints. Thus, we next examine the performance of the considered algorithms when only a small subset of the clients participate in every communication and learning round. Hence, we simulate (7) when the server chooses only a subset of the clients, e.g., $|\mathcal{S}_k| \in \{4, 75, 90\}$. We also simulate (4)-(5) when $|\mathcal{S}_k| = 4$. We present the corresponding learning curves in Fig. 1(b). We observe that, when only a small subset of the clients participate in each FL round, (4)-(5) exhibit poor performance at the presence of link noise. In addition, (7), which exhibited good performance in the previous experiment, fails to converge. It appears to diverge due to error accumulation even when the majority of the clients participate in every FL round, e.g., $|\mathcal{S}_k| \in \{75, 90\}$. Therefore, it is evident that (7) cannot cope with noise in the communication links when only a small number of clients are selected during each iteration.

In our last experiment, we evaluate the performance of the proposed RERCE-Fed algorithm at the presence of noise in the communication links while incorporating random scheduling for communication efficiency. We simulate RERCE-Fed when $|\mathcal{S}_k| \in \{4, 10, 20\}$. We present the corresponding learning curves in Fig. 1(c). We observe that the proposed algorithm exhibits robustness against communication noise/error despite even when a small portion of the clients participate in every FL round. It is also clear that there is a trade-off between $|\mathcal{S}_k|$ and NMSD. Moreover, as the number of participating clients increases, the convergence rate increases. As the number of the participating clients increases, their number becomes less important, i.e., by setting the number of the participating clients to $|\mathcal{S}_k| \geq 10$, the performance is close to when all clients participate. Therefore, a desired performance can be attained with a relatively low number of clients participating at every iteration. This means, using the proposed algorithm, it is possible to achieve accurate model estimates in FL while making efficient use of the available communication resources, even when the communication links are imperfect. The proposed algorithm also delivers an effective trade-off between estimation accuracy and convergence rate on one side and communication resource utilization on the other. This trade-off can be easily governed by controlling the number of clients that participate in FL at every iteration. The participation rate need not necessarily be uniform. That is, depending on resource availability or conditions of the communication links, different numbers of clients may be summoned for FL at different iterations.

6. CONCLUSIONS

We developed a resource-efficient FL algorithm that has improved robustness against noise/error in communication links. To motivate the developed algorithm, we considered a weighted least-squares regression problem. To achieve the robustness, we proposed to combine the last two global model updates and send them together alongside eliminating the dual model update performed at each participating edge device. The proposed algorithm, called RERCE-Fed, ensures that clients receive a less corrupted global model update from the server even when the server uses random scheduling to achieve communication efficiency. We proved the convergence of RERCE-Fed in the mean sense at the presence of link noise. We also verified the desirable performance of RERCE-Fed via simulations, particularly, its robustness against additive communication link noise in comparison to existing related algorithms. In future work, we will analyze the mean-square performance of RERCE-Fed and consider applying it to different applications with potentially non-linear/non-quadratic or non-convex objective functions. We will also study the case when different numbers of clients may participate in different iterations of FL.

7. REFERENCES

[1] X. Wang, C. Wang, X. Li, V. C. M. Leung, and T. Taleb, "Federated deep reinforcement learning for internet of things with decentralized cooperative edge caching," *IEEE Internet Things J.*, vol. 7, no. 10, pp. 9441–9455, Oct. 2020.

[2] Y. Zhao, J. Zhao, L. Jiang, R. Tan, D. Niyato, Z. Li, L. Lyu, and Y. Liu, "Privacy-preserving blockchain-based federated learning for iot devices," *IEEE Internet Things J.*, vol. 8, no. 3, pp. 1817–1829, Feb 2021.

[3] H. B. McMahan, E. Moore, D. Ramage, S. Hampson, and B. A. Y. Arcas, "Communication-efficient learning of deep networks from decentralized data," in *Proc. Int. Conf. Artif. Intell. and Stat.*, Apr. 2017, pp. 1273–1282.

[4] V. Smith, C. Chiang, M. Sanjabi, and A. S. Talwalkar, "Federated multi-task learning," *Proc. Advances in Neural Info. Process. Syst.*, vol. 30, Jan. 2017.

[5] Q. Yang, Y. L., T. Chen, and Y. Tong, "Federated machine learning: Concept and applications," *ACM Transactions on Intelligent Systems and Technology*, vol. 10, no. 2, pp. 1–19, Feb. 2019.

[6] Z. Zhao, C. Feng, W. Hong, J. Jiang, C. Jia, T. Q. S. Quek, and M. Peng, "Federated learning with non-iid data in wireless networks," *" IEEE Trans. Wireless Commun.*, vol. 21, no. 3, pp. 1927–1942, Mar. 2022.

[7] T. Li, A. K. Sahu, A. Talwalkar, and V. Smith, "Federated learning: Challenges, methods, and future directions," *IEEE Signal Process. Mag.*, vol. 37, no. 3, pp. 50–60, May 2020.

[8] W. Y. B. Lim, N. C. Luong, D. T. Hoang, Y. Jiao, Y. Liang, Q. Yang, D. Niyato, and C. Miao, "Federated learning in mobile edge networks: A comprehensive survey," *IEEE Commun. Surveys Tutorials*, vol. 22, no. 3, pp. 2031–2063, Apr. 2020.

[9] T. Gafni, N. Shlezinger, K. Cohen, Y. C. Eldar, and H. V. Poor, "Federated learning: A signal processing perspective," *IEEE Signal Process. Mag.*, vol. 39, no. 3, pp. 14–41, May 2022.

[10] B. Hitaj, G. Ateniese, and F. Perez-Cruz, "Deep models under the gan: information leakage from collaborative deep learning," in *Proc. ACM SIGSAC Conf. Comput. Commun. Security*, 2017, pp. 603–618.

[11] C. Fung, C. J. Yoon, and I. Beschastnikh, "Mitigating sybils in federated learning poisoning," *arXiv preprint arXiv:1808.04866*, Aug. 2018.

[12] V. C. Gogineni, S. Werner, Y. Huang, and A. Kuh, "Communication-efficient online federated learning strategies for kernel regression," *IEEE Internet Things J.*, pp. 1–1, Nov 2022.

[13] ——, "Communication-efficient online federated learning framework for nonlinear regression," in *Proc. IEEE Int. Conf. Acoust. Speech Signal Process.*, May 2022, pp. 5228–5232.

[14] ——, "Decentralized graph federated multitask learning for streaming data," in *Proc. 56th Annu. Conf. Inf. Sci. Syst.*, Apr. 2022, pp. 101–106.

[15] S. Wang, T. Tuor, T. Salonidis, K. K. Leung, C. Makaya, T. He, and K. Chan, "Adaptive federated learning in resource constrained edge computing systems," *IEEE J. on Selected Areas in Commun.*, vol. 37, no. 6, pp. 1205–1221, Mar. 2019.

[16] S. Samarakoon, M. Bennis, W. Saad, and M. Debbah, "Federated learning for ultra-reliable low-latency v2v communications," in *Proc. IEEE Global Commun. Conf.*, Feb. 2018, pp. 1–7.

[17] T. Nishio and R. Yonetani, "Client selection for federated learning with heterogeneous resources in mobile edge," in *Proc. IEEE Int. Conf. on Commun.*, Jul. 2019, pp. 1–7.

[18] S. Zhou and G. Y. Li, "Commun.-efficient ADMM-based federated learning," *arXiv preprint arXiv:2110.15318*, Jan. 2021.

[19] F. Ang, L. Chen, N. Zhao, Y. Chen, W. Wang, and F. R. Yu, "Robust federated learning with noisy communication," *IEEE Trans. Commun.*, vol. 68, no. 6, pp. 3452–3464, Mar. 2020.

[20] S. Zheng, C. Shen, and X. Chen, "Design and analysis of uplink and downlink communications for federated learning," *IEEE J. Sel. Areas Commun.*, vol. 39, no. 7, pp. 2150–2167, Jul. 2021.

[21] M. M. Amiri and D. Gündüz, "Federated learning over wireless fading channels," *IEEE Trans. Wireless Commun.*, vol. 19, no. 5, pp. 3546–3557, May 2020.

[22] H. Guo, A. Liu, and V. K. N. Lau, "Analog gradient aggregation for federated learning over wireless networks: Customized design and convergence analysis," *IEEE Internet Things J.*, vol. 8, no. 1, pp. 197–210, Jan. 2021.

[23] M. M. Amiri, D. Gündüz, S. R. Kulkarni, and H. V. Poor, "Convergence of federated learning over a noisy downlink," *IEEE Trans. Wireless Commun.*, vol. 21, no. 3, pp. 1422–1437, Mar. 2022.

[24] X. Wei and C. Shen, "Federated learning over noisy channels," in *n Proc. IEEE Int. Conf. Commun.*, Jun. 2021, pp. 1–6.

[25] ——, "Federated learning over noisy channels: Convergence analysis and design examples," *IEEE Trans. Cognitive Commun. and Networking*, vol. 8, no. 2, pp. 1253–1268, Jun. 2022.

[26] T. Sery, N. Shlezinger, K. Cohen, and Y. C. Eldar, "COTAF: Convergent over-the-air federated learning," in *Proc. IEEE Global Commun. Conf.*, Dec. 2020, pp. 1–6.

[27] I. D. Schizas, A. Ribeiro, and G. B. Giannakis, "Consensus in ad hoc wsns with noisy links—part I: Distributed estimation of deterministic signals," *IEEE Trans. Signal Process.*, vol. 56, no. 1, pp. 350–364, Jan. 2008.