COMMUNICATION-EFFICIENT ONLINE FEDERATED LEARNING FRAMEWORK FOR NONLINEAR REGRESSION

Vinay Chakravarthi Gogineni[⋆], *Stefan Werner*[⋆], *Yih-Fang Huang*[†], *Anthony Kuh*[‡]

[⋆]Dept. of Electronic Systems, Norwegian University of Science and Technology-NTNU, Norway
[†]Dept. of Electrical Engineering, University of Notre Dame, Notre Dame, IN, USA
[‡]Dept. of Electrical and Computer Engineering, University of Hawaii, Hawaii, USA
E-mails: {vinay.gogineni, stefan.werner}@ntnu.no, huang@nd.edu, kuh@hawaii.edu

ABSTRACT

Federated learning (FL) literature typically assumes that each client has a fixed amount of data, which is unrealistic in many practical applications. Some recent works introduced a framework for online FL (Online-Fed) wherein clients perform model learning on streaming data and communicate the model to the server; however, they do not address the associated communication overhead. As a solution, this paper presents a partial-sharing-based online federated learning framework (PSO-Fed) that enables clients to update their local models using continuous streaming data and share only portions of those updated models with the server. During a global iteration of PSO-Fed, non-participant clients have the privilege to update their local models with new data. Here, we consider a global task of kernel regression, where clients use a random Fourier features-based kernel LMS on their data for local learning. We examine the mean convergence of the PSO-Fed for kernel regression. Experimental results show that PSO-Fed can achieve competitive performance with a significantly lower communication overhead than Online-Fed.

***Index Terms*—** Online federated learning, energy-efficiency, partial-sharing, kernel least mean square, random Fourier features.

1. INTRODUCTION

Federated learning (FL) [1–3] has emerged as an appealing distributed learning framework that allows a network of edge devices to train a global model without revealing local data to others. Among the features that make federated learning (FL) stand out from typical distributed learning, the four most significant are as follows. First, the local data of each client device are not independent and identically distributed (i.e., non-IID) and unbalanced in the amount [4, 5]. Besides that, devices will not disclose the local data to the server or any other client during the training process. Second, federated learning was originally designed to use data collected on battery-constrained or low-performance devices, which also tend to have a low memory capacity [6]. The memory should, therefore, not be depleted by local learning. Third, clients frequently go offline or have limited bandwidth or expensive connections [7, 8]. Lastly, the reliability of clients in federated learning is questionable [9, 10]. Clients with malicious intent may attempt to degrade the global model's reliability. In this paper, we are primarily concerned with reducing communication overhead.

One of the most popular FL methods is federated averaging (FedAvg) [11]. The workflow of FedAvg is as follows: At the start of each global round, a random fraction of clients receive a copy of the global model from the server (generally, clients are selected uniformly, but various methods may be used [12–14]). Using the global model and local data, the selected client then performs multiple iterations of local learning and sends the updated local model to the server. The server then aggregates these updated local models to build a new global model, and the process repeats. Even though FedAvg minimizes communication overhead by executing multiple local updates at each selected client before communicating to the server, its learning accuracy largely depends on the number of epochs (a critical factor in deciding the communication interval) performed at each client [11, 15].

As we can see from the workflow of FL, the participating clients have to communicate the model back and forth with the server in each global iteration. Furthermore, the FL framework generally takes hundreds or thousands of iterations to finalize the global model, and the size of a modern machine learning model is on the order of a billion. The FL framework would have to deal with this enormous communication overhead if utilized. Various solutions have been explored in the literature for reducing the communication overhead associated with FL. Among these, communication-mitigated federated learning (CMFL) [16] discards the irrelevant updates from clients by checking the alignment between the local and global updates tendency. A couple of procedures have been proposed in [17] for reducing uplink communication overhead. The first one is the structured update: clients update the model in a restricted space parametrized with fewer variables. The other one is sketch update: clients update a full model and then compress it using a combination of 1-bit quantization, random rotations, and subsampling before sending it to the server. While sketch updates reduce communication overhead, they are time-consuming and incur additional complexity for clients. By discarding unimportant client updates for global learning (i.e., when numerous model parameters remain unchanged), structured communication reduction for federated learning (FedSCR) [18] can reduce the communication overhead.

The FL approaches outlined above assume a fixed amount of training data at each client, which is impractical in many real-life scenarios, e.g., in wireless communications. Instead, clients may have access to new data or a stream of data during the training [19]. Recently, in [20], the concept of online federated learning (Online-Fed) is discussed; however, no concrete mathematical equations have been given. In Online-Fed, the clients perform online learning on a continuous stream of local data while the server aggregates model parameters received from the clients. On the other hand, the asynchronous online federated learning framework (ASO-Fed) [7]

The Research Council of Norway supported this work.

focused mainly on learning a global model from asynchronous client updates. However, these frameworks have not addressed the communication overhead associated with them.

In this paper, we present an energy-efficient online federated learning framework, namely, partial-sharing-based online federated learning (PSO-Fed), wherein each client updates its local model using continuous streaming data and then shares just a portion of the updated model parameters with the server. In contrast to Online-Fed, PSO-Fed permits non-participant clients to update their local models when they access new data during global iteration. To demonstrate the efficacy of PSO-Fed, we considered nonlinear regression in a non-IID setting. For this, we employ random Fourier features-based kernel LMS (RFF-KLMS) [21–23] to perform the nonlinear regression task locally at each client. In addition, mean convergence analysis of PSO-Fed is provided for these settings. Finally, we perform numerical experiments on synthetic non-IID data, and our results confirm that PSO-Fed achieves competitive performance at very low communication overhead compared to Online-Fed.

2. PROBLEM FORMULATION AND ALGORITHM DESCRIPTION

In this section, we first introduce the Online-Fed in the context of kernel regression. Then, we present an energy-efficient version called PSO-Fed. In the following, we consider a scenario wherein K geographically distributed clients communicate with a global server. At every time instance n, every client k has access to a continuous streaming signal $x_{k,n}$ and associated desired outputs $y_{k,n}$, assumed to be described by the model:

$$y_{k,n} = f(\mathbf{x}_{k,n}) + \nu_{k,n}, \tag{1}$$

where $f(\cdot)$ is a continuous nonlinear model to be estimated collaboratively using clients' data, $\mathbf{x}_{k,n} = [x_{k,n}, \cdots, x_{k,n-L+1}]^{\mathrm{T}}$ is the local data vector of size $L \times 1$, and $\nu_{k,n}$ is the observation noise. For client k, we then define the local optimization function for estimating $f(\cdot)$ as follows:

$$\mathcal{J}_k(\mathbf{w}_k) = E\left[|y_{k,n} - \hat{y}_{k,n}|^2\right], \tag{2}$$

with $\hat{y}_{k,n} = \mathbf{w}_k^{\mathrm{T}} \mathbf{z}_{k,n}$, where the local model parameter vector $\mathbf{w}_k \in \mathbb{R}^D$, is a linear representation of the nonlinear model $f(\cdot)$ in a random Fourier feature (RFF) space of dimension D, and $\mathbf{z}_{k,n} \in \mathbb{R}^D$ being the mapping of $\mathbf{x}_{k,n}$ into RFF space. Cosine, exponential, and Gaussian feature functions [22, 23] can be used to represent $\mathbf{x}_{k,n}$ in the RFF space. Then, the optimization at the global server is

$$\mathcal{J}(\mathbf{w}) = \frac{1}{K} \sum_{k=1}^{K} \mathcal{J}_k(\mathbf{w}). \tag{3}$$

Here, the goal is to find an estimate of global optimal representation of the function $f(\cdot)$ in RFF space, i.e., $\mathbf{w}^\star$ as:

$$\mathbf{w}_n = \min_{\mathbf{w}} \mathcal{J}(\mathbf{w}). \tag{4}$$

2.1. Online-Fed

In each global iteration n, the server selects a subset of clients and share the global model $\mathbf{w}_n$ with them. Thereafter, the selected clients $\forall k \in \mathcal{S}_n$ ($\mathcal{S}_n$ is a set containing selected client indices in global iteration n) run a stochastic gradient descent to solve the local optimization problem $\mathcal{J}_k(\mathbf{w}_k)$ as follows:

$$\mathbf{w}_{k,n+1} = \mathbf{w}_n + \mu\ \mathbf{z}_{k,n}\ \epsilon_{k,n}, \tag{5}$$

where μ is the learning rate and $\epsilon_{k,n} = y_{k,n} - \mathbf{w}_n^{\mathrm{T}} \mathbf{z}_{k,n}$. These clients communicate the updated local models to the server. Then, the server aggregates the received updates as

$$\mathbf{w}_{n+1} = \frac{1}{|\mathcal{S}_n|} \sum_{k \in \mathcal{S}_n} \mathbf{w}_{k,n+1}, \tag{6}$$

where $|\mathcal{S}_n|$ denotes the cardinality of $\mathcal{S}_n$. We see from the workflow of Online-Fed that clients not selected during the nth global iteration do not perform a local model update, despite having access to the local streaming data. Regardless of whether they update, whenever the global server selects them, the latest local model will be replaced by the global model without considering the last update made locally. This issue hinders the performance. Furthermore, the amount of communication taking place at each global iteration is still significant. One solution is to use the concept of structure updates or sketch updates [17]. Our solution to this problem stems from a different approach, namely, partial-sharing concepts [24, 25] that are very attractive for communication-efficient distributed learning.

2.2. PSO-Fed

In the proposed partial-sharing-based online federated learning (PSO-Fed), clients and the server exchange just a fraction of their model parameters in each update round. In order to keep track of the model parameters being exchanged in each communication round, the client and server maintain selection matrices.

To this end, at every global iteration n, the model parameters to be exchanged between clients and the server are specified by a diagonal selection matrix $\mathbf{S}_{k,n}$ of size $D \times D$. On its principal diagonal, $\mathbf{S}_{k,n}$ contains M ones and $D - M$ zeros. In $\mathbf{S}_{k,n}$, the positions of ones specify which local model parameters to be exchanged with the server. As in [24, 25], we can select the M model parameters either stochastically or sequentially. To simplify the implementation, we consider coordinated and uncoordinated partial-sharing. The server assigns the same initial selection matrices to all clients in coordinated partial-sharing (i.e., $\mathbf{S}_{1,0} = \mathbf{S}_{2,0} = \cdots = \mathbf{S}_{K,0} = \mathbf{S}_0$). As a result, all clients exchange the same portion of their local model parameters with the server. On the other hand, in uncoordinated partial-sharing, the server assigns initial selection matrices randomly to clients (i.e., $\mathbf{S}_{1,0} \neq \mathbf{S}_{2,0} \neq \cdots \neq \mathbf{S}_{K,0}$). Both schemes belong to sequential and stochastic partial-sharing families, respectively. For the current global iteration n, the entry selection matrix $\mathbf{S}_{k,n}$ can be obtained via a right circular shift of $\mathbf{S}_{k,n-1}$. In this process, each entry will be exchanged M times over D iterations, so the probability of a specified model parameter being exchanged with the server is $\frac{M}{D}$. With the help of selection matrices, Online-Fed workflow can be alternatively expressed as:

$$\mathbf{w}_{k,n+1} = \mathbf{S}_{k,n}\mathbf{w}_n + (\mathbf{I}_D - \mathbf{S}_{k,n})\mathbf{w}_n + \mu\ \mathbf{z}_{k,n}\ \epsilon_{k,n}, \tag{7a}$$

with $\epsilon_{k,n} = y_{k,n} - (\mathbf{S}_{k,n}\mathbf{w}_n + (\mathbf{I}_D - \mathbf{S}_{k,n})\mathbf{w}_n)^{\mathrm{T}}\mathbf{z}_{k,n}$

$$\mathbf{w}_{n+1} = \frac{1}{|\mathcal{S}_n|} \sum_{k \in \mathcal{S}_n} \mathbf{S}_{k,n+1}\mathbf{w}_{k,n+1} + (\mathbf{I}_D - \mathbf{S}_{k,n+1})\mathbf{w}_{k,n+1}). \tag{7b}$$

Since PSO-Fed limits the exchange of model parameters, the server does not have access to all participating clients' model parameters during the aggregation phase. Similarly, the participating clients do not have access to entire global model parameters; therefore, they will use their previous model parameters in place of the unknown portions. Participating clients use $(\mathbf{I}_D - \mathbf{S}_{k,n})\mathbf{w}_{k,n}$ in

Algorithm 1: `PSO-Fed`. K clients, learning rate μ, set of all clients $\mathcal{S}$, and circular shift variable τ.

Initialization: global model $\mathbf{w}_0$, local model $\mathbf{w}_{k,0}$, RFF space dimension D and selection matrices $\mathbf{S}_{k,0}$, $\forall k \in \mathcal{S}$,

For $n = 1$ to N

The server randomly selects a subset $\mathcal{S}_n$ of K clients and communicate $\mathbf{S}_{k,n}\mathbf{w}_n$ to them,

Client Local Update:

If $k \in \mathcal{S}_n$

$$\epsilon_{k,n} = y_{k,n} - (\mathbf{S}_{k,,n}\mathbf{w}_n + (\mathbf{I}_D - \mathbf{S}_{k,n})\mathbf{w}_{k,n})^{\mathrm{T}}\mathbf{z}_{k,n},$$
$$\mathbf{w}_{k,n+1} = \mathbf{S}_{k,,n}\mathbf{w}_n + (\mathbf{I}_D - \mathbf{S}_{k,n})\mathbf{w}_{k,n} + \mu\ \mathbf{z}_{k,n}\ \epsilon_{k,n},$$

Else

$$\epsilon_{k,n} = y_{k,n} - \mathbf{w}_{k,n}^{\mathrm{T}}\mathbf{z}_{k,n},$$
$$\mathbf{w}_{k,n+1} = \mathbf{w}_{k,n} + \mu\ \mathbf{z}_{k,n}\ \epsilon_{k,n},$$

EndIf

The clients $\forall k \in \mathcal{S}_n$ communicate $\mathbf{S}_{k,n+1}\mathbf{w}_{k,n+1}$ to the server, where $\mathbf{S}_{k,n+1} = \text{circshift}(\mathbf{S}_{k,n}, \tau)$,

Aggregation at the Server:

The server updates the global model as,

$$\mathbf{w}_{n+1} = \frac{1}{|\mathcal{S}_n|}\sum_{k\in\mathcal{S}_n}\mathbf{S}_{k,n+1}\mathbf{w}_{k,n+1} + (\mathbf{I}_D - \mathbf{S}_{k,n+1})\mathbf{w}_n.$$

EndFor

place of $(\mathbf{I}_D - \mathbf{S}_{k,n})\mathbf{w}_n$ and the server uses $(\mathbf{I}_D - \mathbf{S}_{k,n+1})\mathbf{w}_n$ in place of $(\mathbf{I}_D - \mathbf{S}_{k,n+1})\mathbf{w}_{k,n+1}$. The non-participating clients use their previous local models to perform the local learning. The proposed PSO-Fed is summarized in Algorithm 1.

It is important to note that even clients do not take part in all global iterations, the PSO-Fed still permits them to perform local updates as long as they have access to new data. In contrast, as touched upon above, state-of-the-art approaches replace local models with the global model whenever clients are selected for contributing to the model update, making local updates during communication-dormant times futile. It is evident that clients have better control over local learning with PSO-Fed than with current state-of-the-art FL approaches.

3. CONVERGENCE ANALYSIS

In this section, we examine the mean convergence of PSO-Fed. Before proceeding to the analysis, we define the global optimal extended model parameter vector $\mathbf{w}_e^\star = \mathbf{1}_{K+1} \otimes \mathbf{w}^\star$, extended estimated global model parameter vector $\mathbf{w}_{e,n} = \text{col}\{\mathbf{w}_n, \mathbf{w}_{1,n}, \ldots, \mathbf{w}_{K,n}\}$, extended input data matrix $\mathbf{Z}_{e,n} = \text{blockdiag}\{\mathbf{0}, \mathbf{z}_{1,n}, \ldots, \mathbf{z}_{K,n}\}$ and extended observation noise vector $\boldsymbol{\nu}_{e,n} = \text{col}\{\mathbf{0}, \nu_{1,n}, \ldots, \nu_{K,n}\}$, where $\text{col}\{\cdot\}$ and $\text{blockdiag}\{\cdot\}$ represent column-wise stacking operator and block diagonalization operator, respectively. The symbol $\mathbf{1}_{K+1}$ is a $(K+1)\times 1$ column vector with each element taking the value one. From the above definitions, we can write

$$\begin{aligned}\mathbf{y}_{e,n} &= \text{col}\{0, y_{1,n}, y_{2,n}, \ldots, y_{K,n}\} = \mathbf{Z}_{e,n}^{\mathrm{T}}\mathbf{w}_e^\star + \boldsymbol{\nu}_{e,n},\\ \boldsymbol{\epsilon}_{e,n} &= \text{col}\{0, \epsilon_{1,n}, \epsilon_{2,n}, \ldots, \epsilon_{K,n}\} = \mathbf{y}_{e,n} - \mathbf{Z}_{e,n}^{\mathrm{T}}\boldsymbol{\mathcal{A}}_{\mathbf{S},n}\mathbf{w}_{e,n},\end{aligned} \tag{8}$$

with

$$\boldsymbol{\mathcal{A}}_{\mathbf{S},n} = \begin{bmatrix} \mathbf{I}_D & \mathbf{0} & \mathbf{0} & \ldots & \mathbf{0} \\ a_{1,n}\mathbf{S}_{1,n} & \mathbf{I}_D - a_{1,n}\mathbf{S}_{1,n} & \mathbf{0} & \ldots & \mathbf{0} \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ a_{K,n}\mathbf{S}_{K,n} & \mathbf{0} & \mathbf{0} & \ldots & \mathbf{I}_D - a_{K,n}\mathbf{S}_{K,n} \end{bmatrix}, \tag{9}$$

where $a_{k,n} = 1$ if the client $k \in \mathcal{S}_n$, and zero otherwise. Using these definitions, the global recursion of PSO-Fed can be stated as

$$\mathbf{w}_{e,n+1} = \boldsymbol{\mathcal{B}}_{\mathbf{S},n+1}\big(\boldsymbol{\mathcal{A}}_{\mathbf{S},n}\mathbf{w}_{e,n} + \mu\ \mathbf{Z}_{e,n}\ \boldsymbol{\epsilon}_{e,n}\big), \tag{10}$$

where

$$\boldsymbol{\mathcal{B}}_{\mathbf{S},n+1} = \begin{bmatrix} \mathbf{I}_D - \sum\limits_{k\in\mathcal{S}_n}\frac{a_{k,n}}{|\mathcal{S}_n|}\mathbf{S}_{k,n+1} & \frac{a_{1,n}}{|\mathcal{S}_n|}\mathbf{S}_{1,n+1} & \ldots & \frac{a_{K,n}}{|\mathcal{S}_n|}\mathbf{S}_{K,n+1} \\ \mathbf{0} & \mathbf{I}_D & \ldots & \mathbf{0} \\ \vdots & \vdots & \ddots & \vdots \\ \mathbf{0} & \mathbf{0} & \ldots & \mathbf{I}_D \end{bmatrix}. \tag{11}$$

We make the following assumptions to establish the convergence condition for PSO-Fed:

A1: At each client k, the input signal vector $\mathbf{z}_{k,n}$ is drawn from a wide-sense stationary multivariate random sequence with correlation matrix $\mathbf{R}_k = \mathrm{E}[\mathbf{z}_{k,n}\mathbf{z}_{k,n}^{\mathrm{T}}]$.

A2: The noise process $\nu_{k,n}$ is assumed to be zero-mean i.i.d. and independent of all input and output data,

A3: At each client k, the model parameter vector is taken to be independent of input signal vector.

A4: The selection matrices $\mathbf{S}_{k,n}$ are assumed to be independent of any other data; in addition, $\mathbf{S}_{k,n}$ and $\mathbf{S}_{l,m}$ are independent, for all $k \neq l$ and $m \neq n$.

Denoting $\widetilde{\mathbf{w}}_{e,n} = \mathbf{w}_e^\star - \mathbf{w}_{e,n}$, and utilizing the fact that $\mathbf{w}_e^\star = \boldsymbol{\mathcal{B}}_{\mathbf{S},n+1}\boldsymbol{\mathcal{A}}_{\mathbf{S},n}\mathbf{w}_e^\star$ (since $\boldsymbol{\mathcal{B}}_{\mathbf{S},n+1}\mathbf{w}_e^\star = \boldsymbol{\mathcal{A}}_{\mathbf{S},n}\mathbf{w}_e^\star = \mathbf{w}_e^\star$, one can easily prove this result), then from (10), $\widetilde{\mathbf{w}}_{e,n+1}$ can be recursively expressed as

$$\begin{aligned}\widetilde{\mathbf{w}}_{e,n+1} =& \boldsymbol{\mathcal{B}}_{\mathbf{S},n+1}\big(\mathbf{I} - \mu\mathbf{Z}_{e,n}\mathbf{Z}_{e,n}^{\mathrm{T}}\big)\boldsymbol{\mathcal{A}}_{\mathbf{S},n}\widetilde{\mathbf{w}}_{e,n} \\ &- \mu\boldsymbol{\mathcal{B}}_{\mathbf{S},n+1}\mathbf{Z}_{e,n}\boldsymbol{\nu}_{e,n}.\end{aligned} \tag{12}$$

Applying expectation $\mathbb{E}[\cdot]$ on both sides of (12) and using assumptions $\mathbf{A1}-\mathbf{A4}$, we obtain

$$\mathbb{E}[\widetilde{\mathbf{w}}_{e,n+1}] = \mathbb{E}[\boldsymbol{\mathcal{B}}_{\mathbf{S},n+1}]\big(\mathbf{I} - \mu\boldsymbol{\mathcal{R}}_e\big)\mathbb{E}[\boldsymbol{\mathcal{A}}_{\mathbf{S},n}]\mathbb{E}[\widetilde{\mathbf{w}}_{e,n}], \tag{13}$$

where $\boldsymbol{\mathcal{R}}_e = \text{blockdiag}\{\mathbf{0}, \mathbf{R}_1, \mathbf{R}_2, \ldots, \mathbf{R}_K\}$. One can see that $\mathrm{E}\big[\widetilde{\mathbf{w}}_{e,n}\big]$ converges under $\|\mathbb{E}[\boldsymbol{\mathcal{B}}_{\mathbf{S},n+1}]\big(\mathbf{I}-\mu\boldsymbol{\mathcal{R}}_e\big)\mathbb{E}[\boldsymbol{\mathcal{A}}_{\mathbf{S},n}]\| < 1$ for every n, where $\|\cdot\|$ is any matrix norm. Since $\|\mathbb{E}[\boldsymbol{\mathcal{B}}_{\mathbf{S},n+1}]\| = 1$ and $\|\mathbb{E}[\boldsymbol{\mathcal{A}}_{\mathbf{S},n}]\| = 1$, the above convergence condition reduces to $\|\mathbf{I}-\mu\boldsymbol{\mathcal{R}}_e\| < 1$, or, equivalently, $\forall k, i : |1-\mu\lambda_i(\mathbf{R}_k)| < 1$, where $\lambda_i(\cdot)$ is the ith eigenvalue of its argument matrix. After solving the above convergence condition, we finally have following first-order convergence condition:

$$0 < \mu < \frac{2}{\max\limits_{\forall k}\{\max\limits_{\forall i}\{\lambda_i(\mathbf{R}_k)\}\}}. \tag{14}$$

4. NUMERICAL SIMULATIONS

In this section, experimental results are presented to examine the performance of PSO-Fed. Our experiment considers $K = 100$ clients with access to a global server. At every client k, synthetic non-IID input signal $x_{k,n}$ and corresponding observed output are generated so that they are related via the following model:

$$f(\mathbf{x}_{k,n}) = \sqrt{x_{k,1,n}^2 + \sin^2(\pi\, x_{k,4,n})} + \big(0.8 - 0.5\exp(-x_{k,2,n}^2)\big)x_{k,3,n} + \nu_{k,n}. \quad (15)$$

The input signal at each client $x_{k,n}$ was generated by driving a first-order autoregressive (AR) model: $x_{k,n} = \theta_k\, x_{k,n-1} + \sqrt{1-\theta_k^2}\, u_{k,n}$, $\theta_k \in \mathcal{U}(0.2, 0.9)$, where $u_{k,n}$ was drawn from a Gaussian distribution $\mathcal{N}(\mu_k, \sigma_{u_k}^2)$, with $\mu_k \in \mathcal{U}(-0.2, 0.2)$ and $\sigma_{u_k}^2 \in \mathcal{U}(0.2, 1.2)$, respectively (where $\mathcal{U}(\cdot)$ indicates the uniform distribution). The observation noise $\nu_{k,n}$ was taken as zero mean i.i.d. Gaussian with variance $\sigma_{\nu_k}^2 \in \mathcal{U}(0.005, 0.03)$. Using a Cosine feature function, $x_{k,n}$ was mapped into the RFF space whose dimension was fixed to 200. All simulated algorithms were set to the same learning rate of 0.75 for each client. The server implemented uniform random selection procedure to select $|\mathcal{S}_n| = 4$ clients in every global iteration n. By calculating the average mean-square error (MSE) on test data after each global iteration n, we evaluated the simulation performance:

$$\text{MSE} = \frac{1}{N_{\text{test}}}\|\mathbf{y}_{\text{test}} - \mathbf{Z}_{\text{test}}^{\text{T}}\mathbf{w}_n\|_2^2, \quad (16)$$

where $\{\mathbf{Z}_{\text{test}}, \mathbf{y}_{\text{test}}\}$ is the test data set (N_{test} examples in total) covering all clients data. In order to perform the nonlinear regression task, the proposed PSO-Fed was simulated for a variety of values of M (number of model parameters exchanged between the server and clients). In addition, we also simulated the Online-Fed for comparative evaluation. The learning curves (i.e., test MSE in dB against the global iteration index n) are obtained by averaging over 500 independent experiments, are shown in Figs. 1a and 1b for coordinated and uncoordinated partial-sharing schemes, respectively. From Fig. 1, the following interesting observations can be made:

1. With PSO-Fed, we can achieve competitive results at a lower communication cost than Online-Fed. Firstly, PSO-Fed exhibits a slower convergence rate but with a similar steady-state MSE as Online-Fed at small values of M (e.g., 1). As M increases to higher values (e.g., 5 and 40), its convergence becomes faster. In summary, PSO-Fed exhibits a similar convergence rate with a minor improvement in steady-state MSE when $M \geq 40$.
2. Because M is much smaller than D, PSO-Fed's communication cost is lower than that of Online-Fed. PSO-Fed behaves the same as Online-Fed when $M = 40$ but only consumes $\frac{1}{5}$ of its communication load. As non-participating clients make updates locally if they have access to new data, partial-sharing alters only part of these locally updated models when they get a chance to communicate with the server. Thus, resulting in improved performance and reduced communication load. It is worth noting that the proposed partial-sharing does not incur any additional computational overhead, unlike sketch updates proposed in [17]. Keeping track of partially-shared parameter indices just requires a little extra memory.
3. Coordinated partial-sharing has a faster initial convergence speed than uncoordinated one, for very small values of M (e.g., 1 in our experiment). In particular, the coordinated scheme preserves the connectedness of clients by allowing the server to aggregate the same entries of the local model parameter vectors. However, both schemes are equally effective for large values of M (e.g., ≥ 5 in our experiment).

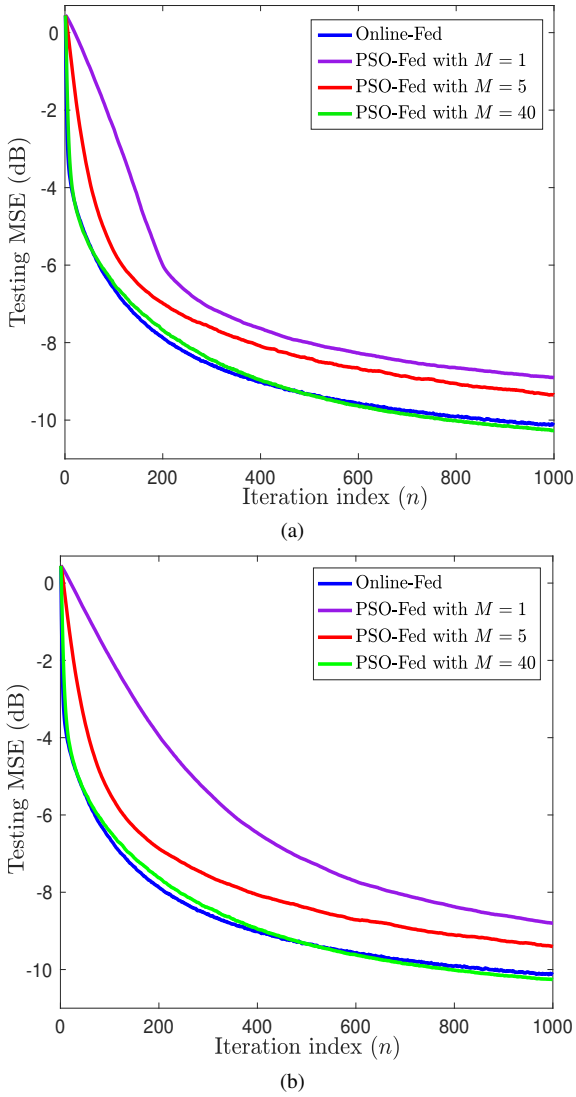

Fig. 1. Performance of PSO-Fed: (a). Coordinated partial-sharing. (b). Uncoordinated partial-sharing.

5. CONCLUSIONS

A communication-efficient framework has been developed for online FL, called PSO-Fed. In PSO-Fed, participating clients exchange a fraction of model parameters with the server, but non-participating clients update their local model if they have access to new data. Thus, the negative effects of partial-sharing have been compensated. PSO-Fed's performance has been demonstrated via kernel regression. The convergence of PSO-Fed has been analyzed under these settings. Simulation results have shown that both coordinated and uncoordinated PSO-Fed algorithms exhibit competitive estimation performance while reducing communication costs compared to Online-Fed.

6. REFERENCES

[1] Q. Yang, Y. Liu, T. Chen, and Y. Tong, "Federated machine learning: Concept and applications," . *ACM Trans. Intell. Syst. Technol.*, vol. 10, no. 2, pp. 2157-6904, Feb. 2019.

[2] 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.

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

[4] V. Smith, C.-K. Chiang, M. Sanjabi, and A. Talwalkar, "Federated multi-task learning," in *Proc. Advances in Neural Info. Process. Syst.*, 2017, pp. 4424–4434.

[5] 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.

[6] Y. Zhou, Q. Ye and J. Lv, "Communication-efficient federated learning with compensated overlap-FedAvg," *IEEE Trans. Parallel Distrib. Syst.*, vol. 33, no. 01, pp. 192-205, 2022.

[7] Y. Chen, Y. Ning, M. Slawski and H. Rangwala, "Asynchronous online federated learning for edge devices with non-IID data," in *Proc. IEEE Int. Conf. Big Data*, 2020, pp. 15-24.

[8] S. Niknam, H. S. Dhillon and J. H. Reed, "Federated learning for wireless communications: Motivation, opportunities, and challenges," *IEEE Commun. Mag.*, vol. 58, no. 6, pp. 46-51, Jun. 2020.

[9] K. Bonawitz, V. Ivanov, B. Kreuter, A. Marcedone, H. B. McMahan, S. Patel, D. Ramage, A. Segal, and K. Seth, "Practical secure aggregation for privacy-preserving machine learning," in *Proc. ACM SIGSAC Conf. Comput. and Commun. Security*, 2017, 1175–1191.

[10] Y. Xuefei, Z. Yanming, and H. Jiankun, "A comprehensive survey of privacy-preserving federated learning: A taxonomy, review, and future directions," *ACM Comput. Surveys*, vol. 54, no. 6, pp. 46-51, Jul. 2021.

[11] 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.*, 2017, vol. 54, pp. 1273-1282.

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

[13] H. T. Nguyen, V. Sehwag, S. Hosseinalipour, C. G. Brinton, M. Chiang and H. Vincent Poor, "Fast-convergent federated learning," *IEEE J. Sel. Areas Commun*, vol. 39, no. 1, pp. 201-218, Jan. 2021.

[14] E. Rizk, S. Vlaski and A. H. Sayed, "Optimal importance sampling for federated learning," in *Proc. IEEE Int. Conf. Acoust., Speech and Signal Process.*, 2021, pp. 3095-3099.

[15] T. Li, A. Kumar Sahu, M. Zaheer, M. Sanjabi, A. Talwalkar and V. Smith, "Federated optimization in heterogeneous networks," in arXiv:1812.06127, 2018, [online]. Available: arXiv: 1812.06127.

[16] L. Wang, W. Wang, and B. Li, "CMFL: Mitigating communication overhead for federated learning," in *Proc. IEEE Int. Conf. Distrib. Comput. Syst.*, 2019, pp. 954–964.

[17] J. Konečný H. B. McMahan, F. X. Yu, P. Richtarik, A. T. Suresh, and D. Bacon, "Federated learning: Strategies for improving communication efficiency," in *Proc. NIPS Workshop on Private Multi-Party Mach. Learn.*, 2016.

[18] X. Wu, X. Yao and C-L. Wang, "FedSCR: Structure-based communication reduction for federated learning," *IEEE Trans. Parallel Distrib. Syst.*, vol. 32, no. 7, pp. 1565-1577, Jul. 2021.

[19] W. U. Bajwa, V. Cevher, D. Papailiopoulos and A. Scaglione, "Machine learning from distributed, streaming Data," *IEEE Signal Process. Mag.*, vol. 37, no. 3, pp. 11-13, May 2020.

[20] A. Kuh, "Real time kernel learning for sensor networks using principles of federated learning," in *Proc. IEEE Int. Conf. Asia Pacific Signal and Info. Process. Assoc.*, 2021 (Accepted).

[21] W. Liu, P. P. Pokharel and J. C. Principe, "The kernel least-mean-square algorithm," *IEEE Trans. Signal Process.*, vol. 56, no. 2, pp. 543-554, Feb. 2008.

[22] A. Rahimi and B. Recht, "Random features for large-scale kernel machines," in *Proc. Adv. Neural Inf. Process. Syst.*, vol. 20, 2007, pp. 1177-1184.

[23] V. C. Gogineni, V. R. M. Elias, W. A. Martins and S. Werner, "Graph diffusion kernel LMS using random Fourier features," in Proc. Asilomar Conf. on Signals, Syst., and Comput., 2020, pp. 1528-1532.

[24] R. Arablouei, S. Werner, Y. F. Huang and K. Doğançay, "Distributed least mean-square estimation with partial diffusion," *IEEE Trans. Signal Process.*, Vol. 62, no. 2, pp. 472–484, Jan. 2013.

[25] R. Arablouei,K. Doğançay, S. Werner and Y. F. Huang, "Adaptive distributed estimation based on recursive least-squares and partial diffusion," *IEEE Trans. Signal Process.*, Vol. 62, no. 14, pp. 3510–3522, Jul. 2014.