DYNAMIC RESOURCE OPTIMIZATION FOR ADAPTIVE FEDERATED LEARNING EMPOWERED BY RECONFIGURABLE INTELLIGENT SURFACES

Claudio Battiloro[1], *Mattia Merluzzi*[3], *Paolo Di Lorenzo*[1,2], *and Sergio Barbarossa*[1,2]

[1]DIET Department, Sapienza University of Rome, Via Eudossiana 18, 00184, Rome, Italy
[2] Consorzio Nazionale Interuniversitario per le Telecomunicazioni (CNIT), Parma, Italy
[3] CEA-Leti, Université Grenoble Alpes, F-38000 Grenoble, France
E-mail: {claudio.battiloro, paolo.dilorenzo, sergio.barbarossa}@uniroma1.it, mattia.merluzzi@cea.fr

ABSTRACT

The aim of this work is to propose a novel dynamic resource allocation strategy for adaptive Federated Learning (FL), in the context of beyond 5G networks endowed with Reconfigurable Intelligent Surfaces (RISs). Due to time-varying wireless channel conditions, communication resources (e.g., set of transmitting devices, transmit powers, bits), computation parameters (e.g., CPU cycles at devices and at server) and RISs reflectivity must be optimized in each communication round, in order to strike the best trade-off between power, latency, and performance of the FL task. Hinging on Lyapunov stochastic optimization, we devise an online strategy able to dynamically allocate these resources, while controlling learning performance in a fully data-driven fashion. Numerical simulations implement distributed training of deep convolutional neural networks, illustrating the effectiveness of the proposed FL strategy endowed with multiple reconfigurable intelligent surfaces.

Index Terms— Adaptive federated learning, Lyapunov optimization, resource allocation, Reconfigurable Intelligent Surfaces.

1. INTRODUCTION

The future of wireless communication systems is to accomodate a plethora of new services for different verticals, on the same network infrastructure [1]. Therefore, beyond 5G networks will have to be extremely flexible and dynamically reconfigurable from the application layer down to the wireless propagation environment. In particular, reconfigurability of the wireless environment is a completely new feature, foreseen to be enabled by the deployment of Reconfigurable Intelligent Surfaces (RIS) [2]. RISs are arrays of backscatters, where each element applies an individual phase-shift (and/or an amplitude and/or a polarization rotation) with which it backscatters an incident wave [3–5], with the aim of creating a controllable reflected beam. The deployment of such a flexible and complex architecture will enable Machine Learning (ML) and Artificial Intelligence (AI) at the edge, a paradigm known as *edge machine learning*. The challenge is that edge ML calls for jointly optimizing inference, training, communication, computation, and control under end-to-end latency, reliability, and learning performance requirements [6–9]. In this context, the exploitation of RISs in this joint design plays a key role, since it can strongly mitigate the presence of poor wireless channel conditions due to mobility, time-varying environment, and blocking events, especially using millimeter wave communications [2, 10].

Related works. Training ML models at the edge mainly relies on *Federated Learning* (FL) [9, 11–15]. In this approach, learning architectures perform (variants of) parallel stochastic gradient descent (SGD) across multiple edge devices, whose intermediate results are aggregated by an Edge Server (ES). FL has several benefits of data privacy, and is empowered by a large amount of device participants with modern powerful processors and low-delay mobile-edge networks. Several works on FL explicitly focus on the optimization of radio resource allocation [16–26]. The authors in [15] study FL and the problem of joint power and resource allocation for ultra-reliable low latency communication in vehicular networks. In [24], the authors propose adapting federated averaging to use a distributed form of Adam optimization along with a compression technique. The work in [27] proposes a FL approach with adaptive and distributed parameter pruning. In [25], the authors propose a joint device scheduling and resource allocation policy to maximize the model accuracy within a given total training time budget. Reference [26] proposes a dynamic user selection scheme to minimize the FL convergence time. Very recently, some works have proposed to enhance the performance of FL tasks by exploiting RISs [28, 29]. In [28], the authors investigate the problem of model aggregation in FL systems aided by multiple RISs, designing the transmit power, selecting the participants in the model uploading process, and tuning the phase shift of the available RISs. Similarly, reference [29] formulates a communication-learning design problem to jointly optimize device selection, over-the-air transceivers, and RIS phase shifts.

Contributions. In this paper, we introduce a novel dynamic optimization framework for adaptive federated learning empowered by RISs, jointly encompassing communication, computation, and learning aspects of the problem. Differently from previous works that mainly focused on static learning (where FL is carried out up to convergence and then the learning process stops), we consider adaptive FL strategies, with the aim of endowing wireless networks with continuous learning and adaptation capabilities. Hinging on Lyapunov stochastic optimization [30], we develop a dynamic resource allocation strategy working at the same time-scale of the FL algorithm, while optimizing on the fly radio parameters (e.g., set of transmitting devices, bits and rates), computation resources (e.g., CPU cycles at devices and at ES) and RISs reflectivity parameters, in order to strike the best trade-off between energy, latency, and performance of the FL task. The method works in a fully data-driven fashion, estimating online the performance (i.e., accuracy and convergence rate) from streaming data. Finally, the proposed strategy is customized to deep convolutional network training, showing the advantages obtained by endowing FL architectures with multiple RISs.

2. SYSTEM MODEL

Let us consider a scenario with N edge devices, K RISs and an AP equipped with an ES. The devices are cooperatively performing a training task aimed at learning a weight vector $\mathbf{w} \in \mathbf{R}^m$. To this

The work of Di Lorenzo and Barbarossa was funded by the H2020 project RISE-6G no. 101017011.

aim, at each time $t \geq 0$, the devices collect batches $\mathcal{B}_t$ of labelled i.i.d. data (i.e., input/output pairs) $(\boldsymbol{x}_{i,t}, y_{i,t}) \in \mathbf{R}^d \times \mathbf{R}$, for all $i = 1, \ldots, N$. Then, assuming that device i has a local loss function $J_i(\mathbf{w}; \boldsymbol{x}_i, y_i)$, whose structure depends on the specific learning task, the goal of FL can be mathematically cast as:

$$\min_{\mathbf{w}} \sum\nolimits_{i=1}^{N} \mathbb{E}\{J_i(\mathbf{w}; \boldsymbol{x}_i, y_i)\}, \tag{1}$$

where the expectation is carried out over the data and/or batch distribution. Now, at each time t, letting $\mathbf{w}_t$ be the instantaneous guess for $\mathbf{w}$, we proceed by optimizing problem (1) using an adaptive stochastic optimizer (e.g. ADAM, Adagrad, etc.). In particular, let us denote $\boldsymbol{g}_{i,t}(\mathbf{w}_t) = \sum_{(\mathbf{x}_{i,t}, y_{i,t}) \in \mathcal{B}_{i,t}} \nabla_{\mathbf{w}} J_i(\mathbf{w}_t; \boldsymbol{x}_{i,t}, y_{i,t})$, for all i, t, to shorten the notation. Then, a rather general gradient-based optimizer recursion reads as:

$$\mathbf{w}_{t+1} = \mathbf{w}_t - \mu \cdot f\left(\sum\nolimits_{i \in \mathcal{S}_t} \boldsymbol{g}_{i,t}(\mathbf{w}_t)\right), \tag{2}$$

where $t \geq 0$, $\mu > 0$ is a step-size, and $\mathcal{S}_t$ is the set of nodes that participate in the optimization at time t; the function $f(\cdot)$ depends on the specific optimizer and applies to the sum of received gradients a time t. To implement (2), the devices belonging to $\mathcal{S}_t$ (to be determined for all t) compute, in parallel, the gradients of the local cost functions and upload them to the AP. Then, the edge server aggregates the local information to compute the new estimate $\mathbf{w}_{t+1}$, which is finally fed back to the devices. To explore the trade-off between power, latency, and performance of FL, following [8], we act on the source encoder of each transmitting device. In particular, we assume the devices transmit dithered quantized versions of $\boldsymbol{g}_{i,t}(\mathbf{w}_t)$ [31], encoding local gradients into $m \cdot b_{i,t}$ bits, where $b_{i,t}$ is chosen at each t from a discrete set $\mathcal{C}_i$, i.e., $b_{i,t} \in \mathcal{C}_i$.

2.1. RIS-enhanced communications

We consider a system endowed with K passive RISs, each one composed of M reflecting elements. Each element is characterized by a complex reflection coefficient $v_{k,l,t} = \theta_{k,l,t} e^{j\phi_{k,l,t}}$, where $\theta_{k,l,t} \in \{0, 1\}$ (i.e., the l-th reflective element of RIS k is active or not at time t), and $\phi_{k,l,t} \in \left\{\dfrac{2n\pi}{2^{b_r}}\right\}_{n=0}^{2^{b_r}-1}$ (i.e., the phase of each element is quantized using b_r bits) [4]. Equivalently, we have

$$v_{k,l,t} \in \mathcal{R} = \left[0, \left\{e^{j\frac{2n\pi}{2^{b_r}}}\right\}_{n=0}^{2^{b_r}-1}\right], \quad \forall k, l, t. \tag{3}$$

Let us denote $\boldsymbol{v}_{k,t} = [v_{k,1,t}, \ldots, v_{k,M,t}]^T$ and $\boldsymbol{v}_t = \{\boldsymbol{v}_{k,t}\}_{k=1}^{K}$. Then, assuming a Single Input Single Output (SISO) communication system, and denoting by $p_{i,t}$ the transmission power of user i, the RIS-aided uplink transmission rate between user i and the AP is

$$R_{i,t} = B_i \log_2\left(1 + \frac{h_{i,t}(\boldsymbol{v}_t) p_{i,t}}{N_0 B_i}\right), \tag{4}$$

where B_i is the bandwidth assigned to user i, N_0 is the noise power spectral density, and $h_{i,t}(\boldsymbol{v}_t)$ is the RIS-dependent channel coefficient given by [32]:

$$h_{i,t}(\boldsymbol{v}_t) = \left| h_{i,t}^a + \sum\nolimits_{k=1}^{K} \boldsymbol{h}_{i,k,t}^T \operatorname{diag}(\boldsymbol{v}_{k,t}) \, \boldsymbol{z}_{i,k,t}^a \right|^2, \tag{5}$$

where $h_{i,t}^a$ represents the direct channel coefficient between user i and the AP; whereas, $\boldsymbol{h}_{i,k,t} \in \mathbb{C}^{M \times 1}$ and $\boldsymbol{z}_{i,k,t}^a \in \mathbb{C}^{M \times 1}$ are vectors containing all the channel coefficients between user i and RIS k elements, and between RIS k elements and the AP, respectively.

2.2. Latency and power consumption

In this paragraph, we evaluate the latency and the power consumption needed to implement one iteration of the algorithm in (2). In particular, there are four main components to be taken into account.

(i) *Local processing:* At the i-th device, the latency and the power necessary to compute the local gradient $\boldsymbol{g}_{i,t}(\mathbf{w}_t)$ are given by $L_{i,t}^{loc} = \frac{B_t J_i}{f_{i,t}}$ and $p_{i,t}^c = \gamma_i (f_{i,t})^3$, respectively, where J_i denotes the number of CPU cycles needed to compute the local gradient from one data unit, $B_t = |\mathcal{B}_t|$, f_i, t is the CPU frequency of device i, and γ_i is the effective switched capacitance of the i-th processor [33].

(ii) *Uplink communication:* Denoting by $R_{i,t}$ the uplink data rate, the latency needed to upload the local gradients to the ES reads as $L_{i,t}^u = \frac{m \cdot b_{i,t}}{R_{i,t}}$. Then, inverting (4), the transmit power consumption is given by: $p_{i,t} = \frac{B_i N_0}{h_{i,t}(\boldsymbol{v}_t)} \left[\exp\left(\frac{R_{i,t} \ln 2}{B_i}\right) - 1\right]$.

(iii) *Edge server processing:* At the ES, the latency necessary to produce the global estimate in (2) is given by $L_t^s = \frac{C|\mathcal{S}_t|}{f_t^s}$, where C is the number of CPU cycles necessary to perform the single step of the gradient-based algorithm for each device, and f_t^s is the CPU frequency of the server. This operation entails a power $p_t^s = \gamma_s (f_t^s)^3$, with γ_s denoting the effective switched capacitance of the ES.

(iv) *Downlink communication:* It is necessary to send the global estimate back to the agents. For simplicity, it is assumed to be given and not optimized in this work.

To have synchronous training updates, we need to consider the maximum among communication delays of all transmitting devices in the overall latency that, at a given time t, reads as:

$$L_t = \max_{i \in \mathcal{S}_t} \left\{L_{i,t}^{loc} + L_{i,t}^u\right\} + L_t^r. \tag{6}$$

Similarly, the overall power consumption for the FL task at time t is:

$$p_t^{\text{tot}} = \sum\nolimits_{i \in \mathcal{S}_t} \left(p_{i,t} + p_{i,t}^c\right) + p_t^s. \tag{7}$$

3. RIS-AIDED ADAPTIVE FEDERATED LEARNING

The aim of this paper is to jointly allocate radio (i.e., set of transmitting devices, powers, quantization bits and RISs reflectivity parameters) and computation (i.e., CPU cycles at devices and at server) resources to minimize the long-term average system power consumption in (7), with constraints on the average learning performance and the average latency in (6). Let G_t and α_t be task-dependent learning performance and convergence rate metrics, respectively, which will be explained in the sequel (see, e.g., (9)). Then, the problem can be cast as:

$$\begin{aligned}
&\min_{\boldsymbol{\Psi}_t} \quad \lim_{t \to \infty} \frac{1}{t} \sum\nolimits_{\tau=0}^{t-1} \mathbb{E}\left\{p_\tau^{\text{tot}}\right\} \\
&\text{subject to} \qquad (a) \ \lim_{t \to \infty} \frac{1}{t} \sum_{\tau=0}^{t-1} \mathbb{E}\left\{L_\tau\right\} \leq \overline{\mathrm{L}}; \qquad\qquad (8) \\
&(b) \ \lim_{t \to \infty} \frac{1}{t} \sum_{\tau=0}^{t-1} \mathbb{E}\{G_\tau\} \geq \overline{\mathrm{G}}; \qquad (c) \ \lim_{t \to \infty} \frac{1}{t} \sum_{\tau=0}^{t-1} \mathbb{E}\{\alpha_\tau\} = \overline{\alpha}, \\
&\left.\begin{array}{l}
b_{i,t} \in \mathcal{C}_i, \ \ \forall i \in \mathcal{S}_t, t; \quad R_i^{\min} \leq R_{i,t} \leq R_{i,t}^{\max}, \ \ \forall i \in \mathcal{S}_t, \ t; \\
f_i^{\min} \leq f_{i,t} \leq f_i^{\max}, \ \ \forall i \in \mathcal{S}_t, \ t; \qquad v_{k,l,t} \in \mathcal{R}, \ \ \forall k, l, t; \\
B_t \in \mathcal{B}, \ \ \forall t; \qquad f^{s,\min} \leq f_t^s \leq f^{s,\max}, \ \ \forall t;
\end{array}\right\} \mathcal{X}_t
\end{aligned}$$

where $\boldsymbol{\Psi}_t = [\boldsymbol{v}_t, \{b_{i,t}\}_{i \in \mathcal{S}_t}, \{R_{i,t}\}_{i \in \mathcal{S}_t}, \{f_{i,t}\}_{i \in \mathcal{S}_t}, f_t^s, B_t]$, and the expectations are taken with respect to the random channel states, whose statistics are supposed to be unknown. The constraints of (8) have the following meaning: (a) the average latency of training iterations does not exceed a predefined value $\overline{\mathrm{L}}$; (b) the average performance metric G_t has to be greater or equal to predefined value $\overline{\mathrm{G}}$; (c) the average convergence rate is constrained to be equal to $\overline{\alpha}$. Also, some constraints in $\mathcal{X}_t$ impose that the batch size B_t can take values only from the discrete set $\mathcal{B}$, the number of quantization bits $\{b_{i,t}\}_{i\in\mathcal{S}_t}$ can take values only from a finite set $\mathcal{C}_i$, and the phase shifts $v_{k,l,t}$ of the RISs obey to (3). Finally, the last constraints in $\mathcal{X}_t$ impose instantaneous bounds on the resource variables $\{R_{i,t}\}_{i\in\mathcal{S}_t}, \{f_{i,t}\}_{i\in\mathcal{S}_t}, f_t^s$.

In several practical cases (e.g., neural network training), we do not have a closed-form expression for G_t and α_t. Therefore, we propose to exploit an online mechanism that estimates the learning performance and the convergence rate in a totally data-driven fashion. We assume that either the ES is provided with a validation set $\mathcal{T}$ or, in the absence of a validation set, the agents can sense an additional batch $\mathcal{T}$ of data at each time-slot, compute their local learning perfomance and send it (one scalar) to the server for the computation of the overall learning perfomance. Then, two task-dependent functions $\widehat{G}_t$ and $\widehat{\alpha}_t$ are introduced to measure online the unknown G_t and α_t, respectively. As an example, if we consider a classification task, the validation (or batch) accuracy and its moving average with length 2κ can be used to estimate learning performance and convergence rate as:

$$\widehat{G}_t = \frac{1}{|\mathcal{T}|}\sum\nolimits_{y\in\mathcal{T}} \mathbb{I}(\widehat{y}_t = y), \quad \widehat{\alpha}_t = \frac{1}{\kappa}\sum_{\tau=t-\kappa}^{\kappa-1}(\widehat{G}_\tau - \widehat{G}_{\tau-1}) \tag{9}$$

where $\widehat{y}_t$ is the prediction for data unit y at time-slot t.

3.1. Algorithmic design via stochastic optimization

We now introduce a method to transform Problem (8) into a stability problem, building on the tools of stochastic Lyapunov optimization [30].In particular, to deal with the long-term constraints (a)-(c), we introduce three *virtual queues*, respectively associated with them:

$$Z_{t+1} = \max\Big\{0, Z_t + \epsilon_z\left(L_t - \overline{\mathrm{L}}\right)\Big\}, \tag{10}$$

$$Q_{t+1} = \max\Big\{0, Q_t + \epsilon_q\left(\overline{\mathrm{G}} - \widehat{G}_t\right)\Big\} \tag{11}$$

$$Y_{t+1} = [Y_t + \epsilon_{y,t}\left(\widehat{\alpha}_t - \overline{\alpha}\right)]\cdot\mathbb{I}\left(\widehat{G}_t \geq \overline{\mathrm{G}}\right), \tag{12}$$

where $\epsilon_z > 0$, $\epsilon_q > 0$ and $\epsilon_{y,t} > 0$ are step-sizes used to speed-up the convergence of the algorithm. The queue evolution defined in (12), including the adaptive step-size $\epsilon_{y,t}$, is motivated by the fact that, if the distribution of the data is stationary, there is no need to overshoot the convergence rate after the target level of learning performance is reached. At the same time, non-stationary behaviors can be detected observing sharp changes in $\widehat{G}_t$, which reactivate the virtual queue Y_t. Interestingly, ensuring the mean-rate stability of the virtual queues in (10)-(12) is equivalent to satisfy the three corresponding constraints [30]. To this aim, letting $\mathcal{U}_t = \frac{1}{2}(Z_t^2 + Q_t^2 + Y_t^2)$, we introduce the following *drif-plus-penalty* function [30], which reads as:

$$\Delta_t^p = \mathbb{E}\{\mathcal{U}_{t+1} - \mathcal{U}_t + V\cdot p_t^{\text{tot}}\big|\,\boldsymbol{\Phi_t}\}, \tag{13}$$

which also incorporates a penalty factor that weights the objective function of (8), with a weighting parameter V. Now, it is possible to prove that minimizing Δ_t^p, if (13) is lower than a finite constant for all t, the virtual queues are mean rate stable, and the optimal solution of (8) is asymptotically reached as V increases [30, Th. 4.8].

Using stochastic optimization arguments [30], we proceed by optimizing a suitable upper-bound of (13), while removing the expectation per each time-slot t. However, the estimates $\widehat{G}_t$ and $\widehat{\alpha}_t$ appearing in (10)-(12) (and, thus, in (13)) are not explicitly related to the number of quantization bits $\{b_{i,t}\}_{i\in\mathcal{S}_t}$ and to the batch size B_t, which must be optimized and adapted to drive the learning performance and the convergence rate. Then, we propose to exploit two surrogate functions, say $\widetilde{G}_t(\{b_{i,t}\}_{i\in\mathcal{S}_t})$ and $\widetilde{\alpha}_t(\{b_{i,t}\}_{i\in\mathcal{S}_t}, B_t)$. The rationale for the selection of the surrogates comes from the assumption that the true performance metrics G_t and α_t typically show a non-increasing behavior with respect to the quantization bits $\{b_{i,t}\}_{i\in\mathcal{S}_t}$ and the batch size B_t (examples will be given in the sequel). In other words, a finer representation of the data typically leads to better learning performance [8, 9].Thus, $\widetilde{G}_t$ and $\widetilde{\alpha}_t$ have only to be non-increasing functions of the quantization bits and the batch size. After some algebra manipulations (omitted due to the lack of space), the method requires to solve the following deterministic problem at each time-slot t:

$$\min_{\boldsymbol{\Psi}_t\in\mathcal{X}_t} \quad Z_t\widetilde{L}_t - \widehat{Q}_t\widetilde{G}_t - \widehat{Y}_t\widetilde{\alpha}_t + V\cdot p_t^{\text{tot}} \tag{14}$$

where $\boldsymbol{\Psi}_t$ is the dynamic set of variables in (8), and $\widetilde{L}_t$ is an upperbound of (6) where the $\max_{i\in\mathcal{S}_t}$ operator is substituted with the summation over all $i\in\mathcal{S}_t$. Because of the structure of the variable set $\boldsymbol{\Psi}_t$, (14) is a mixed-integer nonlinear optimization problem. However, for any given $\{b_{i,t}\}_{i=1}^N$, B_t, and $\boldsymbol{v}_t$ at time t, it is easy to see that (14) is separable into three sub-problems that admit closed form solution for the optimal uplink data rates, the optimal CPU clock frequency of devices, and the optimal CPU clock frequency of the edge server. The expressions, for the optimal parameters are given by (derivations are omitted due to the lack of space):

$$R_{i,t} = \left[\frac{2B_i}{\ln(2)}\,W\!\left(\frac{\ln(2)}{B_i}\sqrt{\frac{Z_t\,m\cdot b_{i,t}\,h_{i,t}(\boldsymbol{v}_t)}{2VN_0}}\right)\right]_{R_i^{\min}}^{R_{i,t}^{\max}} \tag{15}$$

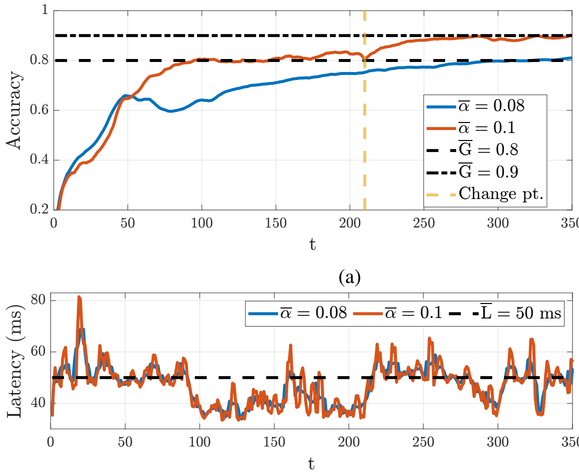
$$f_{i,t} = \left[\left(\frac{Z_tB_tJ_i}{3\gamma_iV}\right)^{\frac{1}{4}}\right]_{f_i^{\min}}^{f_i^{\max}} \qquad f_t^r = \left[\left(\frac{Z_tC\,|\mathcal{S}_t|}{3\gamma_sV}\right)^{\frac{1}{4}}\right]_{f^{r,\min}}^{f^{r,\max}} \tag{16}$$

for all $i\in\mathcal{S}_t$, where $W(\cdot)$ is the principal branch of the Lambert function. In principle, to find the optimal solution of (14) at each time-slot t, one should compute the optimal allocation of edge resources for all possible combinations of $\boldsymbol{v}_t$, $\{b_{i,t}\}_{i=1}^N$ and B_t, evaluate the corresponding objective function using (15) and (16) in (14), and then select the one that yields the lowest value. This approach has a complexity that grows exponentially with N, $\max_i|\mathcal{C}_i|$, $|\mathcal{B}|$, K, M and $|\mathcal{R}|$. Therefore, to find a manageable implementation, in the next paragraph we propose a two-stages greedy method that first optimizes the reflectivity parameters of the RISs $\{\boldsymbol{v}_{k,t}\}_{k=1}^K$ and then, given the RISs configuration, proceeds selecting the batch size B_t and the number of quantization bits $\{b_{i,t}\}_{i=1}^N$.

3.2. Two-stage greedy algorithm for resource optimization

In the first stage, the method greedily selects the RISs coefficients $\{\boldsymbol{v}_{k,t}\}_{k=1}^K$ in order to maximize the following objective:

$$\Delta^R(\{\boldsymbol{v}_{k,t}\}_{k=1}^K) = \sum_{i=1}^N \delta_{i,t}\left|h_{i,t}^a + \sum_{k=1}^K \boldsymbol{h}_{i,k,t}^T\,\mathrm{diag}(\boldsymbol{v}_{k,t})\,\boldsymbol{z}_{i,k,t}^a\right|^2$$



(a)

(b)

Fig. 1: (a) Accuracy vs t. (b) Latency vs t.

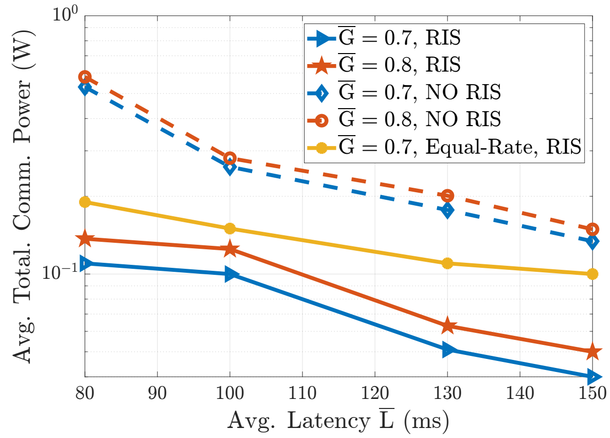


Fig. 2: Total avg. comm. power vs $\overline{L}$ vs for different $\overline{G}$.

where $\delta_{i,t} = \dfrac{1/|h^a_{i,t}|^2}{\sum_{i=1}^N 1/|h^a_{i,t}|^2}$ is a weighting factor aiming at assigning more importance to devices that, without the aid of the RISs, experience worse instantaneous channel conditions. The rationale of this lies on the fact that, having more devices with good channel conditions, increases the degree of freedom in selecting the set of transmitting devices, thus improving the overall performance of the FL task. This greedy procedure experiences a polynomial complexity with respect to K, M and $|\mathcal{R}|$.

In the second stage, for each possible batch size $B_t \in \mathcal{B}$ (the number of selectable batch sizes is assumed to be small, e.g., 3 or 4), the method starts from the empty set of transmitting nodes and iteratively adds the most convenient devices, selecting jointly the best number of quantization bits $\{b_{i,t}\}_{i=1}^N$ and the associated edge resources in (15) and (16). The method keeps adding devices until the resulting value of the objective in (14) decreases, and stops when there is no more incentive in letting other nodes to transmit any bit of information. Finally, the batch size and the associated transmitting set and edge resources that raise the lowest value of the objective function are chosen. Such greedy method hugely reduces the complexity, which becomes polynomial in N, $\max_i\{|\mathcal{C}_i|\}$, $|\mathcal{B}|$.

4. NUMERICAL RESULTS

In this section, we assess the performance of the proposed method, considering a federated learning task aiming at a training a deep convolutional neural network (CNN) classifier. We exploit a CNN architecture made of four convolutional layers with 32, 32, 10 and 10 filters, respectively, with final flatten and dense layers. The loss is the well-known cross-entropy, and the model is trained using a federated ADAM optimizer over the MNIST dataset. We consider a scenario with one AP equipped with an edge server, $N = 9$ devices, 6 of whom having their direct path to the AP attenuated by an obstacle (with an additive pathloss of 5 dB), and one RIS equipped with 1-bit discrete phase shifters. Wireless channels are generated with the tool presented in [32]. We set the radio and computation parameters as follows: $N_0 = -174$ dBm/Hz, $A = 25$, $f_s^{\max} = 3.3$ GHz, $f_i^{\max} = 2.5$ GHz, $R_{i,t}^{\max} = B_i \log_2(1 + h_{i,t}P_i^{\max}/(N_0 B_i))$ where $P_i^{\max}$ = 150 mW for all i; B_i is assigned equally splitting the overall bandwidth equal to 100 MHz among the devices transmitting at time t. Furthermore, the set of quantization bits is $\mathcal{C}_i = [2, 4, 8]$, for all $i = 1, \ldots, N$. The ADAM step-size is set to 0.001, with forgetting factors $\beta_1 = 0.9$, and $\beta_2 = 0.99$. For this experiment, we use the performance estimate $\widehat{G}_t$ and $\widehat{\alpha}_t$ as in (9) with $K = 10$. Also, as a surrogate function for the accuracy metric, we exploit: $\widetilde{G}_t = \sum_{i\in\mathcal{S}_t} \sigma(b_{i,t} - \text{Median}\{\mathcal{C}_i\})$, where $\sigma(\cdot)$ is the logistic sigmoid function, and Median$\{\cdot\}$ represents the median value. Regarding the convergence rate, we use instead the surrogate $\widetilde{\alpha}_t = B_t \sum_{i\in\mathcal{S}_t} b_{i,t}$. Finally, the set of batch sizes is $\mathcal{B} = [1, 3, 7]$. As a first result, in Fig. 1 (a), we illustrate the temporal behavior of the estimated accuracy of the FL algorithm, obtained for different values of the learning rate $\overline{\alpha}$, fixing the accuracy to $\overline{G}$ = 0.8; also, at time slot 210, we change the accuracy requirement from $\overline{G} = 0.8$ to $\overline{G} = 0.9$ for the curve with $\overline{\alpha} = 0.1$, introducing a level of non-stationarity. Then, in Fig. 1 (b), we show the instantaneous latency required by the proposed FL strategy to perform one iteration, together with the latency constraint $\overline{L} = 50$ ms. As we can notice from Fig. 1, the proposed method is able to obtain the desired learning performance, while controlling the convergence rate, and satisfying the required latency constraint. Furthermore, the method is able to react promptly to changes in the accuracy requirement, exhibiting powerful learning and adaptation capabilities in a fully data-driven fashion. Finally, in Fig. 2, we show the total average uplink transmission power expenditure versus the average latency $\overline{L}$, for different values of average accuracy $\overline{G}$, comparing the cases where RISs are exploited or not and a baseline given by an equal-rate (chosen to respect the latency and the accuracy constraints) policy with all the agents always transmitting. As expected, from Fig. 2, the trade-off gets worse imposing a stricter requirement on the accuracy, due to the larger power (and number of bits) necessary to obtain the target performance. Also, we can see the gain obtained thanks to the presence of the RIS in the FL task, and the superior performance of the proposed method w.r.t. the baseline.

5. CONCLUSIONS

In this paper, we proposed an online strategy for adaptive federated learning empowered by reconfigurable intelligent surfaces. The method dynamically minimizes the power expenditure of the system, while guaranteeing target learning performance and latency constraints. The approach builds on stochastic Lyapunov optimization, dynamically and jointly optimizing radio and computation resources, learning parameters, and RISs phase shits. Numerical results illustrate the advantages obtained by the proposed strategy.

6. REFERENCES

[1] E. Calvanese Strinati et al., “6G: The next frontier: From holographic messaging to artificial intelligence using subterahertz and visible light communication,” *IEEE Vehicular Technology Magazine*, vol. 14, no. 3, pp. 42–50, Sep. 2019.

[2] E. Calvanese Strinati et al., “Wireless environment as a service enabled by reconfigurable intelligent surfaces: The RISE-6G perspective,” *Proc. of EUCNC 6G Summit*, Porto, Portugal, 8-11 June 2021.

[3] M. Di Renzo et al., “Smart radio environments empowered by reconfigurable ai meta-surfaces: An idea whose time has come,” *EURASIP Journal on Wireless Commun. and Networking*, vol. 1, pp. 1–20, May, 2019.

[4] P. Mursia, V. Sciancalepore, A. Garcia-Saavedra, L. Cottatellucci, X. Costa-Pérez, and D. Gesbert, “RISMA: Reconfigurable intelligent surfaces enabling beamforming for iot massive access,” *IEEE Journal on Selected Areas in Communications*, 2020.

[5] Hongliang Zhang, Boya Di, Lingyang Song, and Zhu Han, “Reconfigurable intelligent surfaces assisted communications with limited phase shifts: How many phase shifts are enough?,” *IEEE Transactions on Vehicular Technology*, vol. 69, no. 4, pp. 4498–4502, April, 2020.

[6] J. Park, S. Samarakoon, M. Bennis, and M. Debbah, “Wireless network intelligence at the edge,” *Proc. of the IEEE*, vol. 107, no. 11, pp. 2204–2239, November 2019.

[7] Jiasi Chen and Xukan Ran, “Deep learning with edge computing: A review,” *Proceedings of the IEEE*, vol. 107, no. 8, pp. 1655–1674, 2019.

[8] M. Merluzzi, P. Di Lorenzo, and S. Barbarossa, “Wireless edge machine learning: Resource allocation and trade-offs,” *IEEE Access*, vol. 9, pp. 45377–45398, 2021.

[9] C. Battiloro, P. Di Lorenzo, and M. Merluzzi, “Lyapunov-based optimization of edge resources for energy-efficient adaptive federated learning,” *available on TechRxiv, 2021*.

[10] M. Di Renzo et al., “Smart radio environments empowered by reconfigurable intelligent surfaces: How it works, state of research, and the road ahead,” *IEEE Journal on Sel. Areas in Comm.*, vol. 38, no. 11, pp. 2450–2525, November, 2020.

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

[12] Jakub Konečnỳ, H Brendan McMahan, Daniel Ramage, and Peter Richtárik, “Federated optimization: Distributed machine learning for on-device intelligence,” *arXiv preprint arXiv:1610.02527*, 2016.

[13] T. Li, Anit K. Sahu, M. Zaheer, M. Sanjabi, A. Talwalkar, and V. Smith, “Federated optimization in heterogeneous networks,” *preprint arXiv:1812.06127*, 2018.

[14] Jakub Konečnỳ, Brendan McMahan, and Daniel Ramage, “Federated optimization: Distributed optimization beyond the datacenter,” *arXiv preprint arXiv:1511.03575*, 2015.

[15] S. Samarakoon, M. Bennis, W. Saad, and M. Debbah, “Distributed federated learning for ultra-reliable low-latency vehicular communications,” *IEEE Trans. on Communications*, vol. 68, no. 2, pp. 1146–1159, Febraury 2020.

[16] T. Chen, Q. Ling, and G. B. Giannakis, “Learn-and-adapt network resource allocation,” in *Proc. of IEEE SPAWC*, Sapporo, Japan, 3-6 June, 2017, pp. 1–5.

[17] A. S. Bedi and K. Rajawat, “Asynchronous incremental stochastic dual descent algorithm for network resource allocation,” *IEEE Trans. on Signal Processing*, vol. 66, no. 9, pp. 2229–2244, May, 2018.

[18] S. Wang et al., “Adaptive federated learning in resource constrained edge computing systems,” *IEEE Journal on Sel. Areas in Comm.*, vol. 37, no. 6, pp. 1205–1221, June 2019.

[19] M. Chen et al., “A joint learning and communications framework for federated learning over wireless networks,” *IEEE Trans. on Wireless Communications*, vol. 20, no. 1, pp. 269–283, January 2021.

[20] N. H Tran, W. Bao, A. Zomaya, N. M. NH, and C. S. Hong, “Federated learning over wireless networks: Optimization model design and analysis,” in *Proc. of IEEE INFOCOM*, Paris, France, 18-20 September 2019, pp. 1387–1395.

[21] Z. Ma et al., “Adaptive batch size for federated learning in resource-constrained edge computing,” *IEEE Trans. on Mobile Computing, Early Access*, 2021.

[22] L. U. Khan et al., “Federated learning for edge networks: Resource optimization and incentive mechanism,” *IEEE Communications Magazine*, vol. 58, no. 10, pp. 88–93, October 2020.

[23] Z. Yang, M. Chen, W. Saad, C. S. Hong, and M. Shikh-Bahaei, “Energy efficient federated learning over wireless communication networks,” *IEEE Trans. on Wireless Communications*, vol. 20, no. 3, pp. 1935–1949, March 2021.

[24] J. Mills and G. Hu, J.and Min, “Communication-efficient federated learning for wireless edge intelligence in IoT,” *IEEE Internet of Things Journal*, vol. 7, no. 7, pp. 5986–5994, 2020.

[25] W. Shi, S. Zhou, Z. Niu, M. Jiang, and L. Geng, “Joint device scheduling and resource allocation for latency constrained wireless federated learning,” *IEEE Trans. on Wireless Communications*, vol. 20, no. 1, pp. 453–467, Jan 2021.

[26] M. Chen, H. V. Poor, W. Saad, and S. Cui, “Convergence time optimization for federated learning over wireless networks,” *IEEE Trans. on Wireless Communications*, vol. 20, no. 4, pp. 2457–2471, April 2021.

[27] Y. Jiang et al., “Model pruning enables efficient federated learning on edge devices,” *arXiv:1909.12326*, 2020.

[28] Wanli Ni, Yuanwei Liu, Zhaohui Yang, Hui Tian, and Xuemin Shen, “Federated learning in multi-ris aided systems,” *Computing Research Repository, arXiv2010.13333*, October, 2020.

[29] Hang Liu, Xiaojun Yuan, and Ying-Jun Angela Zhang, “Joint communication-learning design for ris-assisted federated learning,” in *Proc. of IEEE ICC Workshops*, Montreal, Canada, 14-23 June, 2021, pp. 1–6.

[30] M. J. Neely, *Stochastic Network Optimization with Application to Communication and Queueing Systems*, Morgan and Claypool, 2010.

[31] S. P. Lipshitz, R. A Wannamaker, and J. Vanderkooy, “Quantization and dither: A theoretical survey,” *Journal of the audio engineering society*, vol. 40, no. 5, pp. 355–375, May 1992.

[32] E. Basar and I. Yildirim, “SimRIS channel simulator for reconfigurable intelligent surface-empowered communication systems,” in *2020 IEEE Latin-American Conference on Communications*, 2020, pp. 1–6.

[33] T. Burd and R. Brodersen, “Processor design for portable systems,” *J. VLSI Sig. Process. Syst.*, vol. 13, no. 2-3, pp. 203–221, Aug. 1996.