Lyapunov-Based Optimization of Edge Resources for Energy-Efficient Adaptive Federated Learning

Claudio Battiloro, *Graduate Student Member, IEEE*, Paolo Di Lorenzo, *Senior Member, IEEE*, Mattia Merluzzi, *Member, IEEE*, and Sergio Barbarossa, *Fellow, IEEE*

***Abstract*—The aim of this paper is to propose a novel dynamic resource allocation strategy for energy-efficient adaptive federated learning at the wireless network edge, with latency and learning performance guarantees. We consider a set of devices collecting local data and uploading processed information to an edge server, which runs stochastic gradient-based algorithms to perform continuous learning and adaptation. Hinging on Lyapunov stochastic optimization tools, we dynamically optimize radio parameters (e.g., set of transmitting devices, transmit powers, bits, and rates) and computation resources (e.g., CPU cycles at devices and at server) in order to strike the best trade-off between power, latency, and performance of the federated learning task. The framework admits both a model-based implementation, where the learning performance metrics are available in closed-form, and a data-driven approach, which works with online estimates of the learning performance of interest. The method is then customized to the case of federated least mean squares (LMS) estimation, and federated training of deep convolutional neural networks. Numerical results illustrate the effectiveness of our strategy to perform energy-efficient, low-latency, adaptive federated learning at the wireless network edge.**

***Index Terms*—Federated learning, Lyapunov stochastic optimization, edge learning, resource allocation.**

I. Introduction

In THE last few years, with the advent of 5G (and beyond) systems, communication networks are evolving from a pure communication framework to service enablers in several different sectors (including *verticals*), such as Industry 4.0, Internet of Things (IoT), autonomous driving, remote surgery, etc. [1]–[5]. As key enablers of this vision, machine learning (ML) and artificial intelligence will be largely exploited in future wireless communication networks, in order to build an effective complex system able to learn and dynamically adapt to the evolving network landscape [6]. Indeed, the advent of a new breed of intelligent devices and high-stake applications foreseen in beyond 5G have sparked a huge interest in distributed, low-latency and reliable ML, calling for a novel system design coined *edge machine learning*, in which: (i) training data is unevenly distributed over a large number of edge devices (including phones, cameras, vehicles, and drones); (ii) every edge device has access to a tiny fraction of the data and training is carried out collectively and distributively; (iii) the inference process is performed on the edge devices, requiring not only high learning accuracy and reliability, but also a very short response time necessary for autonomous decision making in highly dynamic environments. However, differently from cloud-based ML that has virtually infinite computing resources, edge ML is a nascent research field whose system design is entangled with communication and on-device resource constraints (e.g., energy and computing power). Moreover, the process of decentralized training involves a large number of devices that are interconnected over wireless links, hindering learning and adaptation due to communications under poor channel conditions. As such, enabling edge ML introduces novel research problems in terms of jointly optimizing inference, training, communication, computation, and control under end-to-end latency, reliability, and learning performance requirements [7]–[10].

Related works: Training ML models at the edge mainly relies on *federated learning* (FL) [11]–[19]. These 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. The work in [11] provides a comprehensive survey on FL algorithms and introduces various challenges, problems, and solutions for enhancing FL effectiveness. In [12], the authors develop two update methods to reduce the uplink communication costs for FL. The work in [15] presents a practical update method for a deep FL algorithm and conduces an extensive empirical evaluation for different FL models. The authors in [18] study FL and the problem of joint power and resource allocation for ultra-reliable low latency communication in vehicular networks. The work in [19] develops a new approach to minimize the computing and transmission delay for FL algorithms. Other works on FL explicitly focus on the optimization of radio resource allocation [20]–[33]. In [20], the authors propose a control algorithm that determines the best trade-off between

Manuscript received 30 September 2021; revised 30 March 2022; accepted 20 June 2022. Date of publication 28 June 2022; date of current version 15 February 2023. This work was supported in part by the H2020 EU-Taiwan Project 5G-CONNI under Grant AMD-861459-3. The work of Mattia Merluzzi was supported by the European Commission through the H2020 Project Hexa-X under Grant 101015956. *(Corresponding author: Claudio Battiloro.)*

Claudio Battiloro, Paolo Di Lorenzo, and Sergio Barbarossa are with the Department of Information Engineering, Electronics, and Telecommunications, Sapienza University of Rome, 00184 Rome, Italy (e-mail: claudio.battiloro@uniroma1.it; paolo.dilorenzo@uniroma1.it; sergio.barbarossa@uniroma1.it).

Mattia Merluzzi is with Univ. Grenoble Alpes, CEA, Leti, 38000 Grenoble, France (e-mail: mattia.merluzzi@cea.fr).

Digital Object Identifier 10.1109/TGCN.2022.3186879

local update and global parameter aggregation to minimize the loss function under a given resource budget. In [21], the authors propose energy-efficient strategies for bandwidth allocation and scheduling to enable latency-constrained FL. The work in [22] proposes a joint learning and wireless resource allocation framework to optimize the FL performance. In [23], the authors characterize how the computation and communication latencies of edge devices affect the trade-offs between energy consumption, learning time, and accuracy of the FL task. The work in [25] studies the relationship between batch size and convergence rate to alleviate the negative impact of synchronization barrier through adaptive batch size during model training in the FL paradigm. In [26], the authors model the interaction between a global server and the participating devices for federated learning via a Stackelberg game to motivate the participation of the devices in the federated learning process. In [27], the authors provide an optimization problem whose goal is to minimize the total energy consumption of the system under a latency constraint; to solve the problem, an iterative algorithm is proposed where, at every step, closed-form solutions for time allocation, bandwidth allocation, power control, computation frequency, and learning accuracy are derived. The work in [28] propose a federated deep-reinforcement-learning-based cooperative edge caching framework which enables base stations (BSs) to cooperatively learn a shared predictive model by considering the first-round training parameters of the BSs as the initial input of the local training, and then uploads near-optimal local parameters to the BSs to participate in the next round of global training. In [29], the authors propose adapting federated averaging to use a distributed form of Adam optimization along with a compression technique. The work in [30] proposes a FL approach with adaptive and distributed parameter pruning, which adapts the model size during FL to reduce both communication and computation overhead and minimize the overall training time, while maintaining a similar accuracy as the original model. In [31], the authors consider two transmission protocols for edge devices to upload model parameters to edge server, based on non orthogonal multiple access and time division multiple access, respectively. Under both protocols, they minimize the total energy consumption at all edge devices over a particular finite training duration subject to a given training accuracy, by jointly optimizing the transmission power and rates at the edge devices for uploading model parameters and their central processing unit frequencies for local update. In [32], the authors propose a joint device scheduling and resource allocation policy to maximize the model accuracy within a given total training time budget for latency constrained wireless FL. The work in [34] analyzes how to design dynamic FL in mobile edge networks that optimally chooses the number of selected clients and the number of local iterations in each training round to minimize the total cost while ensuring convergence. Reference [33] proposes a dynamic user selection scheme to minimize the FL convergence time. In [35] authors propose two bandwidth allocation schemes to maximize the number of active clients under latency and bandwidth constraints. The work in [36] focuses on the design and analysis of physical layer quantization and transmission methods for wireless FL. In [37] authors propose a strategy that allocates different aggregation weights to different clients based on the heterogeneous quantization errors of all clients. The work in [38] propose a strategy for joint allocation of wireless resources and quantization bits across the clients to minimize the quantization errors while making the clients have the same transmission outage probability. Finally, some works exploited Lyapunov optimization for federated learning [39]–[41]. In [39] authors propose to statically optimize agents' schedule and power allocation to minimize the global learning loss under an energy consumption constraint. In [40] authors propose to optimize admitted data proportions, load balancing, training scheduling and numerical accuracy to minimize a unified cost function under stability of the data queue constraint. Finally, the work in [41] proposes to optimize agents' schedule to minimize long-term average model exchange time under a fairness constraint.

Contributions: The goal of this paper is to introduce a novel dynamic optimization framework for *adaptive federated learning*, which jointly encompasses communication, computation, and learning aspects of the problem. Differently from previous works that mainly focused on a static learning task (where FL is carried out up to convergence and then the learning process stops), here we consider adaptive FL strategies, with the aim of endowing wireless networks with continuous learning, adaptation, and tracking capabilities [42]. Hinging on Lyapunov stochastic optimization [43], we develop a dynamic resource allocation strategy that works at the same time-scale of the gradient-based algorithm, while optimizing on the fly radio parameters (e.g., set of transmitting devices, transmit powers, bits and rates) and computation resources (e.g., CPU cycles at devices and at edge server) in order to strike the best trade-off between energy, latency, and performance (e.g., convergence rate, accuracy or mean-squared error) of the adaptive FL task. The proposed method encompassing jointly communication, computation, and learning aspects of FL represents the main distinctive difference with respect to the previous approaches hinging on Lyapunov optimization [39]–[41]. In this paper, particular emphasis is devoted to the definition and the online control of proper performance metrics in both a model based scenario, where the metrics are known in closed-form, and in a data-driven case, where performance must be inferred online from data. In both cases, the method is able to adaptively minimize the average power needed for the FL task, while ensuring guaranteed latency and learning performance. Finally, the proposed strategy is customized to adaptive federated Least Mean Squares (LMS) estimation and deep convolutional neural network training. Part of this work was presented in the preliminary conference paper [44], which is here largely extended in terms of theory, models, and numerical results. Due to the upcoming convergence of communication and learning in beyond 5G networks, it is fundamental to merge these aspects with a mathematically formal analysis of both communication and computation power consumption and delays, as well as learning performance metrics in terms of accuracy of the learning task, rate of convergence and adaptation. This mathematical analysis, also exploited in the proposed online solution, represents the main

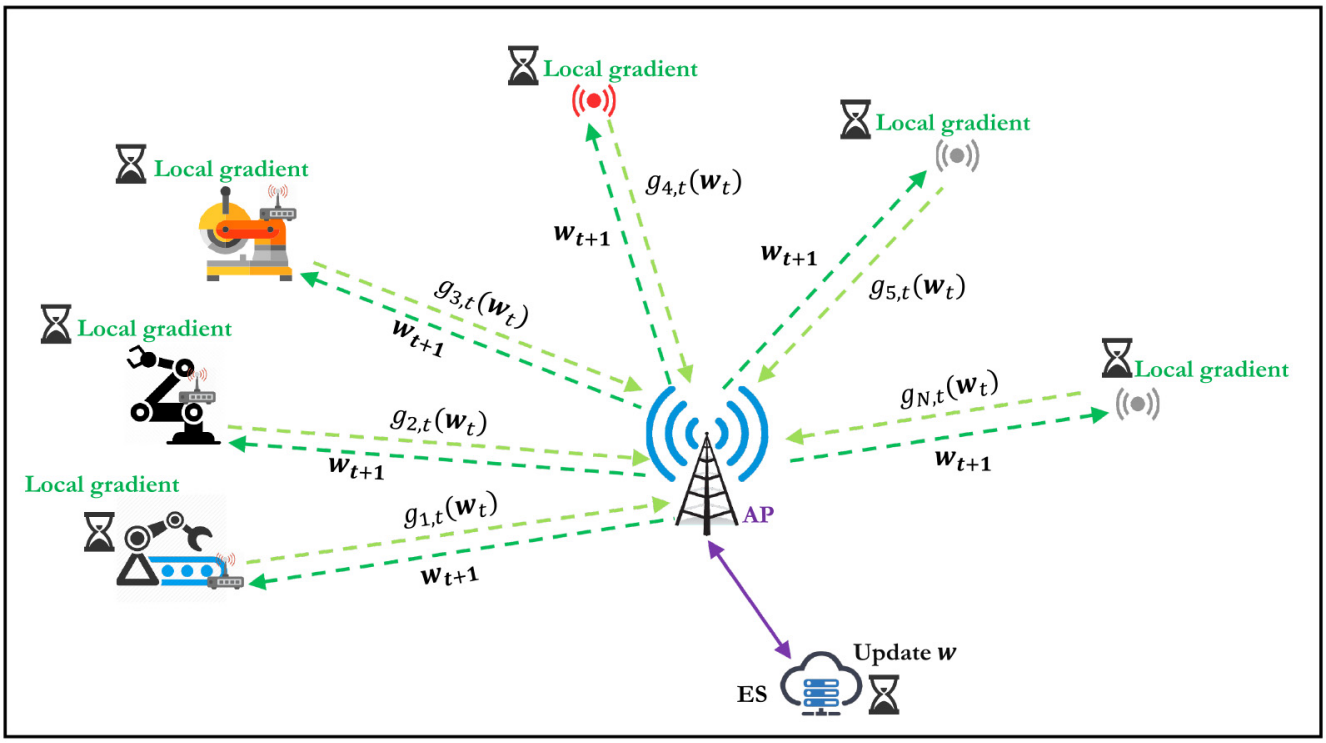


Fig. 1. Federated learning scenario.

contribution of our work with respect to the current state of the art previously presented.

Notation: Scalar, column vector, and matrix variables are respectively indicated by plain letters a (A), bold lowercase letters $\boldsymbol{a}$, and bold uppercase letters $\boldsymbol{A}$. $\mathbb{I}(\cdot)$ denotes the indicator function; a_{ij} is the (i, j)-th element of $\boldsymbol{A}$, $\mathbf{I}$ is the identity matrix, and $\mathbf{1}_N$ ($\mathbf{0}_N$) is the $N \times 1$ vector of all ones (zeros). diag$\{\boldsymbol{a}\}$ denotes a diagonal matrix having vector $\boldsymbol{a}$ on its main diagonal. $\mathbb{E}\{\cdot\}$ denotes the expectation operator, $\mathrm{Tr}\{\cdot\}$ denotes the matrix trace operator, and $\lambda_{\min}\{\cdot\}$ represents the minimum eigenvalue. Other specific notation is defined along the paper.

II. System Model

In this section, we present the mathematical model used to describe the FL algorithm and its performance, the overall latency, and the system power consumption. Let us consider a scenario with N edge devices and an AP equipped with an edge server, as illustrated in Fig. 1. The devices are cooperatively performing a training task aimed at learning a weight vector $\mathbf{w} \in \mathbf{R}^m$. To this aim, at each time t, the devices collect labelled data (i.e., input/output pairs) given by $(\boldsymbol{x}_{i,t}, y_{i,t}) \in \mathbf{R}^d \times \mathbf{R}$, for all $i = 1, \ldots, N$, and $t \geq 0$. Then, assuming that device i has a local loss function $J_i(\mathbf{w}; \boldsymbol{x}_{i,t}, y_{i,t})$, whose structure depends on the specific learning task, the goal of FL can be mathematically cast as:

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

where the expectation is carried out over the data distribution. Now, at each time t, letting $\mathbf{w}_t$ be the instantaneous guess for the weight vector $\mathbf{w}$, we proceed by optimizing problem (1) using an adaptive stochastic gradient descent procedure [42]. In particular, let us denote

$$\boldsymbol{g}_{i,t}(\mathbf{w}_t) = \nabla_{\mathbf{w}} J_i(\mathbf{w}_t; \boldsymbol{x}_{i,t}, y_{i,t}),$$

for all i, t, to shorten the notation. Then, the adopted SGD recursion reads as:

$$\mathbf{w}_{t+1} = \mathbf{w}_t - \mu \sum_{i \in \mathcal{S}_t} \boldsymbol{g}_{i,t}(\mathbf{w}_t), \tag{2}$$

where $t \geq 0$, $\mu > 0$ is a (sufficiently small) step-size parameter, and $\mathcal{S}_t$ is the set of nodes that participate to the optimization at time t. In the considered FL scenario, the algorithm in (2) can be implemented in two ways. The straightforward one requires that, at each time t, the edge devices belonging to $\mathcal{S}_t$ (to be determined for all t) compute in parallel the gradients of the local cost functions (i.e., $\boldsymbol{g}_{i,t}(\mathbf{w}_t)$) 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. An example of the data exchange required for this implementation is illustrated in Fig. 1. The second implementation of algorithm (2) requires instead that, at each time t, multiple edge devices compute one step of gradient-based algorithms on the current model using local data, and then the server takes a weighted average of the resulting models. In particular, device i evaluates a local estimate $\boldsymbol{\psi}_{i,t}$ given by:

$$\boldsymbol{\psi}_{i,t} = \mathbf{w}_t - \mu |\mathcal{S}_t| \, \boldsymbol{g}_{i,t}(\mathbf{w}_t), \qquad \forall i \in \mathcal{S}_t. \tag{3}$$

Then, the edge server aggregates the local information in (3) to compute $\mathbf{w}_{t+1}$ as:

$$\mathbf{w}_{t+1} = \frac{1}{|\mathcal{S}_t|} \sum_{i \in \mathcal{S}_t} \boldsymbol{\psi}_{i,t}. \tag{4}$$

It is straightforward to see that the combination of (3) and (4) is equivalent to the direct SGD implementation in (2). The latter implementation is known as federated averaging (FedAvg), and is typically preferred for privacy reasons [15], since transmitting the gradients might reveal information about the data. However, the first implementation is less sensible to errors introduced by quantization effects, which can be controlled through the choice of the step-size μ. For this reason, following the idea of [10], we will act on the source encoder of each transmitting device, dynamically adapting the quantization level in order to strike the best trade-off between power, latency, and learning performance. Furthermore, adding quantization noise to gradients naturally induces a differential privacy behavior [45], [46]. Thus, in the sequel, we will consider only the direct SGD implementation in (2), thus modeling its energy consumption, latency, and learning performance. Similar expressions hold for the FedAvg implementation in (3)-(4).

A. Dithered Quantization of Uplink Data

We assume that each device i is endowed with a dynamic uniform quantizer that uses $b_{i,t}$ bits to transmit data to the AP at time t. The quantizer used by device i at time t is defined by the following vector mapping $\boldsymbol{q}(\cdot; b_{i,t})$ as:

$$\boldsymbol{q}(\boldsymbol{z}; b_{i,t}) = \left[l_1\Delta_{i,t}, \ldots, l_n\Delta_{i,t}\right]^T = \boldsymbol{z} + \boldsymbol{e}_{i,t}(\boldsymbol{z}), \quad (5)$$

where the entries of $\boldsymbol{z}$, the dynamic quantization step $\Delta_{i,t} > 0$, and the error $\boldsymbol{e}_{i,t}$ satisfy

$$\begin{cases} \Delta_{i,t} = \frac{A}{2^{b_{i,t}}-1}, & \forall i, t, \\ (l_n - 1/2)\Delta_{i,t} \leq z_n \leq (l_n + 1/2)\Delta_{i,t}, & \forall n, i, t, \\ -\frac{\Delta_{i,t}}{2}\mathbf{1}_m \leq \boldsymbol{e}_{i,t}(\boldsymbol{z}) \leq \frac{\Delta_{i,t}}{2}\mathbf{1}_m, & \forall \boldsymbol{z}, i, t, \end{cases} \quad (6)$$

with A denoting the size of input data variation. Conditioned on the input, the quantization error $\boldsymbol{e}_{i,t}(\boldsymbol{z})$ in (5) is deterministic. This induces a correlation among the quantization errors at different times, which may affect the convergence properties of the iterative algorithm (2). To avoid undesired error correlations, we introduce dithering [47], [48]. In particular, the dither added to randomize the quantization effects satisfies a special condition, namely the Schuchman condition, as in subtractively dithered systems [49]. Then, adding to each component z_n a dither noise $\nu_{n,i,t}$ of i.i.d. uniformly distributed random variables on $[-\Delta_{i,t}/2, \Delta_{i,t}/2)$ independent of the input sequence, the resultant error $\epsilon_{n,i,t}$ becomes

$$\epsilon_{n,i,t} = q(z_n + \nu_{n,i,t}; b_{i,t}) - (z_n + \nu_{n,i,t}), \quad (7)$$

which is uniformly distributed over $[-\Delta_{i,t}/2, \Delta_{i,t}/2)$ and independent of z_n. To implement (2), the devices transmit dithered quantized versions of $\boldsymbol{g}_{i,t}(\mathbf{w}_t)$ at time t, which from (7) read as:

$$\boldsymbol{q}(\boldsymbol{g}_{i,t}(\mathbf{w}_t) + \boldsymbol{\nu}_{i,t}; b_{i,t}) = \boldsymbol{g}_{i,t}(\mathbf{w}_t) + \underbrace{\boldsymbol{\nu}_{i,t} + \boldsymbol{\epsilon}_{i,t}}_{\boldsymbol{v}_{i,t}(b_{i,t})}, \quad (8)$$

where $\boldsymbol{v}_{i,t}(b_{i,t})$ is an additive zero mean noise term with covariance $\frac{\Delta_{i,t}^2}{6}\mathbf{I}$. The dithered gradients in (8) are encoded into $m \cdot b_{i,t}$ bits, and transmitted by device i at time t.

B. Performance of Adaptive Federated Learning

In this section, we derive expressions for the learning performance of the proposed FL strategy. Using the dithered quantized gradients (8) in (2), the SGD recursion is given by:

$$\begin{aligned} \mathbf{w}_{t+1} &= \mathbf{w}_t - \mu \sum_{i\in\mathcal{S}_t} \boldsymbol{q}(\boldsymbol{g}_{i,t}(\mathbf{w}_t) + \boldsymbol{\nu}_{i,t}; b_{i,t}) \\ &= \mathbf{w}_t - \mu \sum_{i\in\mathcal{S}_t} \left[\boldsymbol{g}_{i,t}(\mathbf{w}_t) + \boldsymbol{v}_{i,t}(b_{i,t})\right] \\ &= \mathbf{w}_t - \mu \sum_{i\in\mathcal{S}_t} \left[\mathbb{E}\{\boldsymbol{g}_{i,t}(\mathbf{w}_t)\} + \boldsymbol{s}_{i,t}(\mathbf{w}_t, b_{i,t})\right], \end{aligned} \quad (9)$$

where $s_{i,t}(\mathbf{w}_t, b_{i,t})$ is a gradient noise process defined as:

$$\boldsymbol{s}_{i,t}(\mathbf{w}_t, b_{i,t}) = \boldsymbol{g}_{i,t}(\mathbf{w}_t) - \mathbb{E}\{\boldsymbol{g}_{i,t}(\mathbf{w}_t)\} + v_{i,t}(b_{i,t}), \quad (10)$$

for all $i \in \mathcal{S}_t$ and $t \geq 0$, which depends on both data and dithered quantization statistics. Then, from (9), the aim of the analysis is to find expressions for the algorithm's performance in terms of learning accuracy and convergence rate. Let us denote by G_t the performance metric for the learning problem at time slot t. Depending on the specific task, G_t might represent different metrics as, e.g., prediction error or classification accuracy. Also, let us denote by α_t the convergence rate at time slot t. Now, performing a mean-square analysis for the recursion in (10) can be a formidable task for general cost functions. Thus, in the sequel, we consider two different assumptions on the global loss function in (1).

1) Strongly Convex Objective: In this case, we assume that (1) has a favorable structure that helps making the problem mathematically tractable. In particular, letting

$$J(\mathbf{w}) = \sum_{i=1}^{N} \mathbb{E}\{J_i(\mathbf{w}; \boldsymbol{x}_i, y_i)\},$$

we consider the following assumption [42].

Assumption 1: The aggregate cost $J(\mathbf{w})$ is twice differentiable, ν-strongly convex, and its gradient is δ-Lipschitz.

This is the case of many important learning paradigms, spanning from least-mean squares adaptation, support vector machines, logistic regression, and so on [42]. As we will see in the sequel, Assumption 1 is useful to give closed form expressions for the steady-state performance and convergence rate of our federated learning strategy (cf. (13) and (15)), which are in turn important to control the resource allocation of our system in a very efficient manner using Lyapunov optimization (cf. Section III). Furthermore, following similar arguments as in [42, Ch. 5], we consider the following assumption on the gradient noise process affecting (9).

Assumption 2: The gradient noise process in (10) is zero-mean and satisfies

$$\mathbb{E}\left\{\|\boldsymbol{s}_{i,t}(\mathbf{w}_t, b_{i,t})\|^2 | \mathcal{F}_{t-1}\right\} \leq \beta^2\|\mathbf{w}_t - \mathbf{w}^o\|^2 + \sigma_s^2, \quad (11)$$

for all $i = 1, \ldots, N$ and $t \geq 0$, and some β^2 and σ_s^2, where $\mathcal{F}_{t-1}$ is the filtration of the random process $\mathbf{w}_t$ up to time $t-1$, and $\mathbf{w}^o$ is the global minimum of $J(\mathbf{w})$ (which exists under Assumption 1). Assumption 2 ensures boundedness of the second-order moments of the gradient noise in (10), and is instrumental to derive the mean-square performance

of the algorithm in (9). From (9) and (10), it is clear that the performance of the SGD algorithm depends on the set $\mathcal{S}_t$ (i.e., which devices are transmitting) and $\{b_{i,t}\}_{i\in\mathcal{S}_t}$ (i.e., how many bits encode the information transmitted by each device). Of course, the two variables are related to each other as:

$$i \in \mathcal{S}_t \quad \Longleftrightarrow \quad b_{i,t} \neq 0, \tag{12}$$

i.e., a device belongs to the set of transmitting nodes if the number of uploaded bits is different from zero, and viceversa. Thus, the set $\mathcal{S}_t$ is fully known given the quantization bits $\{b_{i,t}\}_{i=1}^N$, which represent the variables to be selected by the resource allocation algorithm. In particular, for a fixed transmission scheme (i.e., $\mathcal{S}_t = \mathcal{S} \Longleftrightarrow b_{i,t} = b_i$ for all t), under Assumptions 1 and 2, for any $0 < \mu < 2\nu/(\delta^2+\beta^2)$, the Mean-square Deviation (MSD) and the Excess Risk (ER) can be expressed as [42, Ch. 5]:

$$\begin{aligned}\mathrm{MSD} &= \lim_{t\to\infty} \mathbb{E}\Big\{\|\mathbf{w}_t - \mathbf{w}^o\|^2\Big\} \\ &= \frac{\mu}{2}\,\mathrm{Tr}\left\{\left(\sum_{i\in\mathcal{S}} \mathbf{H}_i\right)^{-1}\left(\sum_{i\in\mathcal{S}} \mathbf{R}_i(b_i)\right)\right\},\end{aligned} \tag{13}$$

$$\mathrm{ER} = \lim_{t\to\infty} \mathbb{E}\{J(\mathbf{w}_t) - J(\mathbf{w}^o)\} = \frac{\mu}{4}\,\mathrm{Tr}\left\{\sum_{i\in\mathcal{S}} \mathbf{R}_i(b_i)\right\} \tag{14}$$

where $\mathbf{H}_i = \nabla^2_{\mathbf{w}}\mathbb{E}\{J_i(\mathbf{w}^o; \boldsymbol{x}_i, y_i)\}$ is the local Hessian, and

$$\mathbf{R}_i(b_i) = \mathbb{E}\Big\{\boldsymbol{s}_i(\mathbf{w}^o, b_i)\boldsymbol{s}_i(\mathbf{w}^o, b_i)^T|\mathcal{F}_{t-1}\Big\}$$

is the covariance matrix of the local gradient noise in (10), both evaluated at $\mathbf{w}^o$, for all $i = 1, \ldots, N$. Also, we can approximate the convergence rate (i.e., the rate at which the error variance $\mathbb{E}\{\|\mathbf{w}_t - \mathbf{w}^o\|^2\}$ approaches its steady-state region) of the algorithm in (9) by [42, Ch. 5]:

$$\alpha = 1 - 2\mu\,\lambda_{\min}\left(\sum_{i\in\mathcal{S}} \mathbf{H}_i\right). \tag{15}$$

The smaller is α from (15), the higher is the convergence rate. Thus, hinging on (13) and (15), a possible way to define approximate learning performance metrics of (9) at time t is:

$$G_t = \mathrm{MSD}(\mathcal{S}_t, \{b_{i,t}\}_{i=1}^N) \tag{16}$$

$$\alpha_t = \alpha(\mathcal{S}_t, \{b_{i,t}\}_{i=1}^N). \tag{17}$$

Alternatively, the learning accuracy G_t can be defined with respect to the ER in (14). Of course, the expressions (16) and (17) represent only (instantaneous) approximations of the learning performance achieved by (9), considering the values of $\{b_{i,t}\}_{i\in\mathcal{S}_t}$ at a given time-slot t. However, as we will show in the numerical results, an accurate performance prediction can still be achieved, thanks to the fast adaptation capabilities of algorithm (9) in both training and steady-state phases.

2) Non-Convex Objective: The non-convex scenario is typical of many FL tasks involving, e.g., (deep) neural network training. In such a case, we almost never have theoretical performance metrics, which enable reliable prediction of the algorithm's accuracy, following a model-based approach. Thus, we follow an alternative data-driven approach, which involves an online estimation of learning performance, i.e., accuracy and convergence rate, in order to drive the dynamic resource allocation.

In Section III, we will first derive the resource allocation framework assuming to have access to some closed form expression of the performance metrics G_t and α_t, as in (13) and (15). Then, in Section IV, we will extend the proposed framework to handle the case where closed form expressions for G_t and α_t are not generally available and must be inferred online from the data, which is typical of non-convex scenarios. Specific details of the proposed data-driven approach will be provided in Section IV.

C. Latency of SGD Iterations

The latency necessary to perform one SGD iteration at time t, together with the convergence rate in (17), quantify the time the FL algorithm in (9) needs to learn and adapt. In particular, the overall latency of one iteration is composed of four main sources of delay, which vary over time due to availability of resources (radio and computation) and wireless channel states.

(*i*) The *local processing time* to compute the gradient $g_{i,t}(\mathbf{w}_t)$ at the i-th device reads as:

$$L^l_{i,t} = \frac{N^l_i}{f^l_{i,t}}, \qquad i \in \mathcal{S}_t, \tag{18}$$

where N^l_i is the number of CPU cycles necessary to perform this task, and $f^l_{i,t}$ is the CPU frequency of device iy at time t.

(*ii*) The *uplink communication time*, necessary to upload the local gradients to the edge server. Since the i-th device adopts a dithered quantization scheme that encodes local gradients into $m \cdot b_{i,t}$ bits at time ty, this latency term reads as:

$$L^u_{i,t} = \frac{m \cdot b_{i,t}}{R^u_{i,t}}, \qquad i \in \mathcal{S}_t, \tag{19}$$

where $R^u_{i,t}$ is the uplink data rate. In principle, if device iy belongs to $\mathcal{S}_t$, it is going to compute and transmit at time t, incurring in the delays (18) and (19).

(*iii*) The *remote processing time* at the edge server, necessary to produce the global estimate in (9) is given by

$$L^r_t = \frac{O\,|\mathcal{S}_t|}{f^r_t}, \tag{20}$$

where O is the number of CPU cycles necessary to perform one summation (between m-dimensional vectors), the cardinality $|\mathcal{S}_t|$ denotes the number of transmitting nodes, and f^r_t is the CPU frequency of the edge server.

(*iv*) A *downlink communication time*, say $L^d_{i,t}$, $i = 1, \ldots, N$, necessary to send the global estimate $\mathbf{w}_{t+1}$ back to the devices. Here, we assume that the number of bits used to encode downlink data is a fixed value, which is chosen sufficiently large to have a negligible impact on the performance of the algorithm in (9). Also, since our interest is mainly focused on uplink communications, the downlink communication time $L^d_{i,t}$ is assumed to be given and ensured by the AP at any slot; thus, it will not be optimized over time.

Finally, to evaluate the overall latency of each SGD iteration, we need to consider the maximum among communication delays of all transmitting (and receiving) devices in the

overall latency, plus the computation time at the edge server, which at a given time ty reads as:

$$L_t = \max_{i\in\mathcal{S}_t}\Big\{L^l_{i,t} + L^u_{i,t}\Big\} + L^r_t + \max_{i=1,\dots,N}\Big\{L^d_{i,t}\Big\}. \quad (21)$$

As we will show in the sequel, our aim is to keep the average value of L_t in (21), i.e., the average time of SGD iterations, below a given threshold.

D. Power Consumption

In this paragraph, we evaluate the power consumption of the proposed federated learning strategy. We consider two sources of power consumption for each device: local computation and transmission; then, we take into account the power spent for computation at the edge server. At time t, the power spent by device i for local computation is:

$$p^c_{i,t} = \kappa_l (f^l_{i,t})^3, \quad (22)$$

where κ_l is the effective switched capacitance of the processor [50]. Moreover, given the uplink data rate $R^u_{i,t}$, the power spent for uplink transmission is computed by inverting the Shannon formula, thus obtaining:

$$p^u_{i,t} = \frac{B^u_{i,t} N_0}{h^u_{i,t}}\left[\exp\left(\frac{R^u_{i,t}\ln 2}{B^u_{i,t}}\right) - 1\right], \quad (23)$$

where $B^u_{i,t}$ is the bandwidth assigned to device iy at time ty, $h^u_{i,t}$ is the uplink channel power gain, and N_0 is the noise power spectral density. At each time t, we assume that the AP allocates orthogonal frequency channels with pre-allocated bandwidth $B^u_{i,t}$ to all transmitting users (i.e., those belonging to $\mathcal{S}_t$). On the other side, at time ty, the power spent by the edge server for computation of (9) is given by:

$$p^c_{s,t} = \kappa_r (f^r_t)^3, \quad (24)$$

where f^r_t and κ_r are the CPU frequency and the effective switched capacitance of the ES processor, respectively.

In this paper, our goal is to minimize the long-term average of the system power consumption, given by the sum of the devices and ES powers, which reads as:

$$p^{\text{tot}}_t = \sum_{i\in\mathcal{S}_t}\Big(p^u_{i,t} + p^c_{i,t}\Big) + p^c_{s,t}. \quad (25)$$

In the following section, we will formulate the proposed dynamic strategy for wireless network edge optimization, aimed at performing energy-efficient FL with guaranteed latency and learning performance requirements.

III. Dynamic Optimization of Wireless Edge Resources

We can now formulate the problem of dynamic resource allocation for FL. The aim is to find the optimal joint dynamic resource allocation of radio (i.e., the set $\mathcal{S}_t$ of transmitting devices, uplink data rates $\{R^u_{i,t}\}_{i\in\mathcal{S}_t}$, quantization bits $\{b_{i,t}\}_{i\in\mathcal{S}_t}$) and computation (i.e., CPU cycles at devices $\{f^l_{i,t}\}_{i\in\mathcal{S}_t}$ and at the server f^r_t) resources to minimize the long-term average system power consumption in (25), with constraints on the average learning performance in (16)-(17), and the average latency in (21). Then, the dynamic resource allocation problem can be cast as:

$$\begin{aligned}
\min_{\mathbf{\Psi}_t}\ \lim_{t\to\infty}\ & \frac{1}{t}\sum\nolimits_{\tau=0}^{t-1}\mathbb{E}\{p^{\text{tot}}_\tau\} \\
\text{subject to}\ & (a)\ \lim\nolimits_{t\to\infty}\frac{1}{t}\sum_{\tau=0}^{t-1}\mathbb{E}\{L_\tau\} \le \overline{\text{L}}; \\
& (b)\ \lim_{t\to\infty}\frac{1}{t}\sum_{\tau=0}^{t-1}\mathbb{E}\{G_\tau\} \le \overline{\text{G}}; \\
& (c)\ \lim_{t\to\infty}\frac{1}{t}\sum_{\tau=0}^{t-1}\mathbb{E}\{\alpha_\tau\} = \overline{\alpha}; \\
& \left.\begin{aligned} b_{i,t}\in\mathcal{B}_i \quad & \forall i\in\mathcal{S}_t, t; \\ R^{\min}_i \le R^u_{i,t} \le R^{\max}_{i,t}\ & \forall i\in\mathcal{S}_t,\ t; \\ f^{\min}_i \le f^l_{i,t} \le f^{\max}_i\ & \forall i\in\mathcal{S}_t,\ t; \\ f^{r,\min} \le f^r_t \le f^{r,\max}, \quad & \forall t, \end{aligned}\right\}\mathcal{X}_t
\end{aligned} \quad (26)$$

where $\mathbf{\Psi}_t = [\{b_{i,t}\}_{i\in\mathcal{S}_t}, \{R^u_{i,t}\}_{i\in\mathcal{S}_t}, \{f^l_{i,t}\}_{i\in\mathcal{S}_t}, f^r_t]$, and the expectations are taken with respect to the random channel states, whose statistics are supposed to be unknown. The constraints of (26) have the following meaning: (a) the average latency of SGD iterations does not exceed a predefined value $\overline{\text{L}}$ (although more sophisticated probabilistic or instantaneous constraints can be used [51], the average latency constraint (a) relaxes the resource allocation policy, avoiding excessive power consumption or unfeasible solutions in the case of bad channel conditions); (b) the average performance metric G_t does not exceed a predefined value $\overline{\text{G}}$; if G_t represents an accuracy metric, the sign of the constraint should be reversed, i.e., the average accuracy must be greater than a certain target value; (c) the average convergence rate is constrained to be equal to $\overline{\alpha}$; finally, the constraints in $\mathcal{X}_t$ impose that $\{b_{i,t}\}_{i\in\mathcal{S}_t}$ can take values only from a finite set $\mathcal{B}_i$ of discrete quantization bits, and impose instantaneous bounds (e.g., budget constraints, minimum rates and CPU frequencies) on the resource variables $\{R^u_{i,t}\}_{i\in\mathcal{S}_t}, \{f^l_{i,t}\}_{i\in\mathcal{S}_t}, f^r_t$. In the sequel, we introduce a dynamic algorithmic framework to solve the long-term optimization problem (26).

A. Algorithmic Solution via Stochastic Optimization

We now introduce a method to transform (26) into a stability problem, building on the tools of stochastic Lyapunov optimization [43]. In particular, to deal with the long-term constraints (a)-(c), we introduce three *virtual queues*. The first one, used to impose (a), evolves as:

$$Z_{t+1} = \max\big\{0, Z_t + \epsilon_z\big(L_t - \overline{\text{L}}\big)\big\}, \quad (27)$$

where ϵ_z is a positive step-size used to control the convergence speed of the algorithm. The second virtual queue, used to impose constraint (b), read as:

$$Q_{t+1} = \max\big\{0, Q_t + \epsilon_q\big(G_t - \overline{\text{G}}\big)\big\}, \quad (28)$$

with $\epsilon_q > 0$. Finally, the virtual queue associated with constraint (c) is given by:

$$Y_{t+1} = Y_t + \epsilon_y(\alpha_t - \overline{\alpha}), \quad (29)$$

with $\epsilon_y > 0$. Note that virtual queue Y_t reads slightly differently from the others, due to the fact that it is used to impose an equality constraint [43]. Interestingly, ensuring the mean-rate stability of the virtual queues in (27)-(29) is equivalent to satisfy the three corresponding constraints [43]. To this aim, we first define the Lyapunov function as:

$$\mathcal{U}_t = \mathcal{U}(\mathbf{\Phi}_t) = \frac{1}{2}\Big(Z_t^2 + Q_t^2 + Y_t^2\Big), \tag{30}$$

where $\mathbf{\Phi}_t = [Z_t, Q_t, Y_t]$. The Lyapunov function is a measure of the congestion state of the virtual queues, and is fundamental to define the *drift-plus-penalty* function [43]:

$$\Delta_t^p = \mathbb{E}\big\{\mathcal{U}_{t+1} - \mathcal{U}_t + V \cdot p_t^{\text{tot}} \big| \, \mathbf{\Phi}_t\big\}. \tag{31}$$

The drift-plus-penalty function in (31) is the conditional expected change of $\mathcal{U}_t$ over successive slots, with a penalty factor that weights the objective function of (26), with a weighting parameter V. Now, if Δ_t^p is lower than a finite constant for all ty, the virtual queues are stable and the optimal solution of (26) is asymptotically reached as V increases [43, Th. 4.8]. In practical scenarios with finite Vy values, the higher is V, the more importance is given to the objective function, rather than the virtual queue backlogs, thus pushing the solution toward optimality while still guaranteeing stability of the system. Thus, we proceed by minimizing an upper-bound of the drift-plus-penalty in (31), which reads as follows:

$$\begin{aligned}\Delta_t^p \leq \zeta + \mathbb{E}\Big\{ & Z_t\big(L_t - \bar{L}\big) + Q_t\big(G_t - \overline{\mathrm{G}}\big) \\ & + Y_t(\alpha_t - \bar{\alpha}) + V \cdot p_t^{\text{tot}} \Big| \mathbf{\Phi}_t \Big\},\end{aligned} \tag{32}$$

where G_t, α_t and p_t^{tot} are defined in (16), (17) and (25), respectively; whereas, ζ is a positive finite constant. The derivations leading to (32) and the value of ζ can be found in the Appendix. Finally, using stochastic approximation arguments [43], we optimize (32) removing the expectation, thus obtaining the following deterministic problem at each time-slot t (omitting all the constant terms):

$$\begin{aligned}\min_{\mathbf{\Psi}_t} \quad & Z_t L_t + Q_t G_t + Y_t \alpha_t + V \cdot p_t^{\text{tot}} \\ \text{subject to} \quad & \mathbf{\Psi}_t \in \mathcal{X}_t\end{aligned} \tag{33}$$

where $\mathcal{X}_t$ is the instantaneous feasible set of (26). Now, following [43], for a fixed Vy, solving (33) in each time slot guarantees that all virtual queues are mean-rate stable, so that constraints (*a*), (*b*), (*c*) of (26) are met. Furthermore, hinging on the concept of a C-additive approximation [43], if the per-slot solution comes within a finite constant C from the optimum, we have:

$$\limsup_{t\to\infty} \frac{1}{t} \sum_{\tau=0}^{t-1} \mathbb{E}\{p_\tau^{\text{tot}}\} \leq p^{\text{tot,opt}} + \frac{\zeta + C}{V}, \tag{34}$$

where $p^{\text{tot,opt}}$ is the infimum time average power achievable by any policy that meets the required constraints, and ζ is the constant whose expression can be found in the Appendix, see (57). Of course, the higher is C, the higher is the value of V needed to approach this asymptotic optimality, which in this case translates into higher queue backlogs. Based on the above concept, to further decrease complexity, we now solve (33) by replacing L_t with an upper bound $\widetilde{L}_t$ given by

$$\widetilde{L}_t = \sum_{i\in\mathcal{S}_t} \left(\frac{N_i^l}{f_{i,t}^l} + \frac{m \cdot b_{i,t}}{R_{i,t}^u} \right) + \frac{O\,|\mathcal{S}_t|}{f_t^r} + \max_{i=1,\ldots,N} \Big\{ L_{i,t}^d \Big\}, \tag{35}$$

obtained by applying the straightforward upper bound $\max\{x_i\} \leq \sum_{i=1}^N x_i$. Now, because of the structure of $\mathcal{X}_t^i$, (33) is a mixed-integer nonlinear optimization problem, which might be very complicated to solve. However, for any given $\{b_{i,t}\}_{i=1}^N$ at time-slot t, it is easy to see that (33) is separable into three sub-problems that admit closed form solutions for the optimal uplink data rates, the optimal CPU frequency of devices, and the optimal CPU frequency of the edge server, respectively. In the sequel, we present the formulation and the solution of the three sub-problems.

B. Uplink Radio Resource Allocation

The uplink radio resource allocation sub-problem aims at optimizing the transmission rates $\{R_{i,t}^u\}_{i\in\mathcal{S}_t}$ of each transmitting device at time-slot t, once the quantization bits $\{b_{i,t}\}_{i=1}^N$ have been fixed. Of course, if $b_{i,t} = 0$, we have $R_{i,t}^u = 0$. Instead, for any $b_{i,t} > 0$, from (25), (33) and (35), we obtain:

$$\begin{aligned}\min_{\{R_{i,t}^u\}_{i\in\mathcal{S}_t}} & \sum\nolimits_{i\in\mathcal{S}_t} \left(Z_t \frac{m \cdot b_{i,t}}{R_{i,t}^u} + V p_{i,t}^u \right) \\ \text{subject to } & R_i^{\min} \leq R_{i,t}^u \leq R_{i,t}^{\max} \quad \forall i \in \mathcal{S}_t,\end{aligned} \tag{36}$$

where $R_{i,t}^{\max}$ is the maximum rate achievable using the maximum transmitted power, and $R_i^{\min}$ represents the minimum rate that a user should use in the case of transmission. Of course, in (36), there is an intrinsic admission control condition embedded in terms of feasibility. In particular, denoting the set of nodes for which problem (36) is feasible by:

$$\mathcal{A}_t = \{i \in \{1,\ldots,N\} : R_{i,t}^{\max} > R_i^{\min}\}, \tag{37}$$

it clearly holds that the set $\mathcal{S}_t$ of transmitting nodes must be selected as a subset of $\mathcal{A}_t$ ($\mathcal{S}_t \subseteq \mathcal{A}_t$). If problem (36) is feasible, it is also strictly convex with respect to the rates $\{R_{i,t}^u\}_{i\in\mathcal{S}_t}$, and admits a unique closed-form solution. In particular, the Lagrangian function of (36) writes as:

$$\begin{aligned}\mathcal{L}_i^R = \sum\nolimits_{i\in\mathcal{S}_t} \Bigg[& \left(Z_t \frac{m \cdot b_{i,t}}{R_{i,t}^u} + V p_{i,t}^u \right) - \delta_i \Big(R_{i,t}^u - R_i^{\min} \Big) \\ & + \xi_i \Big(R_{i,t}^u - R_{i,t}^{\max} \Big) \Bigg],\end{aligned} \tag{38}$$

where δ_i and ξ_i are the Lagrange multipliers associated with the constraints of (36) over the variable $R_{i,t}^u$, for $i \in \mathcal{S}_t$. Then, the Karush-Kuhn-Tucker (KKT) conditions of the strictly convex problem (36) are given by:

$$\begin{aligned} i)\ & \frac{\partial \mathcal{L}_i^R}{\partial R_{i,t}} = -Z_t \frac{m \cdot b_{i,t}}{(R_{i,t}^u)^2} + \frac{V N_0}{h_{i,t}^u} \exp\left(\frac{R_{i,t}^u \ln(2)}{B_{i,t}^u} \right) \\ & -\delta_i + \xi_i = 0;\end{aligned} \tag{39}$$

$$ii)\, R^u_{i,t} \geq R^{\min}_i, \quad \delta_i \geq 0, \quad \delta_i\left(R^u_{i,t} - R^{\min}_i\right) = 0; \quad (40)$$

$$iii)\, R^u_{i,t} \leq R^{\max}_{i,t}, \quad \xi_i \geq 0, \quad \xi_i\left(R^u_{i,t} - R^{\max}_{i,t}\right) = 0; \quad (41)$$

for all $i \in \mathcal{S}_t$, where we used (23) for $p^u_{i,t}$. Now, exploiting the principal branch of the Lambert function $W(\cdot)$ [52], the KKT conditions (39)-(41) can be solved in closed form as:

$$R^u_{i,t} = \left[\frac{2B^u_{i,t}}{\ln(2)}\, W\left(\frac{\ln(2)}{B^u_{i,t}}\sqrt{\frac{Z_t\, m \cdot b_{i,t}\, h^u_{i,t}}{2\, V N_0}}\right)\right]^{R^{\max}_{i,t}}_{R^{\min}_i} \quad (42)$$

for all $i \in \mathcal{S}_t$, if problem (36) is feasible.

C. Local Computing Resources Allocation

The local computing resource allocation problem aims at optimizing the CPU frequencies $\{f^l_{i,t}\}_{i\in\mathcal{S}_t}$ of the transmitting (and computing) devices. From (33), (35) and (25), for a given time t, it is easy to see how the local computing resource allocation problem decouples over the computing devices and over the iterations within a slot. Thus, we obtain the following sub-problem at each device:

$$\begin{aligned} \min_{f^l_{i,t}} \;& Z_t \frac{N^l_i}{f^l_{i,t}} + V\kappa_l (f^l_{i,t})^3 \\ \text{subject to} \;& f^{\min}_i \leq f^l_{i,t} \leq f^{\max}_i, \end{aligned} \quad (43)$$

for all $i \in \mathcal{S}_t$. Problem (43) is strictly convex and enjoys a simple closed form solution. Indeed, solving the KKT conditions, it is immediate to obtain:

$$f^l_{i,t} = \left[\left(\frac{Z_t N^l_i}{3\kappa_l V}\right)^{\frac{1}{4}}\right]^{f^{\max}_i}_{f^{\min}_i} \quad \text{for all } i \in \mathcal{S}_t. \quad (44)$$

Note that (44) contains implicitly the admission control condition of the radio resource problem (36), i.e., a local computation is needed only if data can be subsequently uploaded to the AP. Thus, if $i \notin \mathcal{S}_t$, then $R^u_{i,t} = 0$ and also $f^l_{i,t} = 0$.

D. Remote CPU Frequencies Allocation

The remote computing resource allocation problem aims at optimizing the CPU frequency f^r_t of the ES. From (33), (35) and (25), for a given $\mathcal{S}_t$ at time t, the remote computing resource allocation problem writes as:

$$\begin{aligned} \min_{f^r_t} \;& Z_t \frac{O\,|\mathcal{S}_t|}{f^r_t} + V\kappa_r (f^r_t)^3 \\ \text{subject to} \;& f^{r,\min} \leq f^r_t \leq f^{r,\max}. \end{aligned} \quad (45)$$

Problem (45) is a strictly convex optimization problem, which enjoys a simple closed form solution. Then, solving the KKT conditions, it is straightforward to see that the optimal remote CPU cycle frequency is given by

$$f^r_t = \left[\left(\frac{Z_t O\,|\mathcal{S}_t|}{3\kappa_l V}\right)^{\frac{1}{4}}\right]^{f^{r,\max}}_{f^{r,\min}}. \quad (46)$$

E. Online Selection of Transmitting Users, Quantization Bits, and Wireless Edge Resources

In Sections III-B–III-D, we have derived the optimal allocation of edge resources as a function of the quantization bits $\{b_{i,t}\}_{i=1}^N$. Thus, exploiting (42), (44), and (46), the objective of (33), say $\widetilde{\Delta}^p_t$, can now be expressed as a function of only $\{b_{i,t}\}_{i=1}^N$, i.e., $\widetilde{\Delta}^p_t = \widetilde{\Delta}^p_t(\{b_{i,t}\}_{i=1}^N)$. In principle, to find the optimal solution of (33) at each time-slot t, one should compute the optimal allocation of edge resources for all possible combinations of $\{b_{i,t}\}_{i=1}^N$, evaluate the corresponding objective function $\widetilde{\Delta}^p_t$ in (33), and then select the one that yields the lowest value. This approach faces a main challenge: Even if for the single choice of $\{b_{i,t}\}_{i=1}^N$ the resource allocation is efficient (cf. (42), (44), and (46)), the overall search procedure has still a complexity that grows exponentially with the number N of devices and the maximum cardinality of the set of quantization bits (i.e., $\max_i |\mathcal{B}_i|$). For this reason, in the sequel we act some simplifications to reduce the complexity of (33), while still achieving good performance. In particular, instead of performing an exhaustive search over all possible combinations, we use an iterative greedy approach that, starting from the empty set of transmitting nodes, iteratively adds the most convenient devices, selecting jointly the best number of quantization bits and the associated edge resources in (42), (44), and (46). The method keeps adding devices from the admissible set $\mathcal{A}_t$ in (37) until the resulting value of the objective $\widetilde{\Delta}^p_t$ in (33) decreases, and stops when there is no more incentive (in terms of reduction of the objective function) in letting other nodes to transmit any bit of information. Of course, if $\mathcal{A}_t$ is empty, the t-th iteration of the FL algorithm in (9) does not take place. Such greedy method drastically reduces the complexity of the procedure, which becomes polynomial in N and $\max_i |\mathcal{B}_i|$. Then, once the resource have been selected, the federated learning algorithm is updated as in (9), and, finally, the virtual queues Z_t, Q_t and Y_t are updated as in (27), (28) and (29), respectively. The overall dynamic optimization procedure for adaptive federated learning is illustrated in Algorithm 1.

Remark 1: Of course, there are no guarantees that the proposed greedy procedure in Algorithm 1 finds the optimal solution of (33), inevitably representing only an approximation of it, so that our approach can be, also in this case, interpreted as a C-additive approximation [43, p. 59], which entails inexact solutions (with bounded error) of the drift-plus-penalty method in (33) at each iteration t, as anticipated in Section III-A. In our case, since the objective and the feasible set of (33) are both bounded for all ty, the proposed greedy approach clearly leads to a valid C-approximation. In Section IV, we will numerically assess the performance of the proposed dynamic resource allocation strategy for adaptive federated learning at the wireless network edge.

IV. Data-Driven Resource Allocation for Adaptive Federated Learning

In the previous section, we have proposed a model-based algorithm for dynamic resource allocation to enable FL at the wireless edge, which exploits closed-form expressions for the

Algorithm 1: Model-Based Dynamic Resource Optimization for Adaptive Federated Learning

for $t \geq 0$ **do**

S1. Observe Z_t, Q_t, Y_t, $\{h^u_{i,t}\}_{i=1}^N$ and compute $\mathcal{A}_t$ in (37).

if $\mathcal{A}_t = \emptyset$ **then**

continue

else

S2. Find $\mathcal{S}_t$, the quantization bits $\{b_{i,t}\}_{i\in\mathcal{S}_t}$, and the edge resources through the greedy procedure:

Set $\mathcal{S}_t = \emptyset$, $\{b_{i,t}\}_{i=1}^N = 0$, $\Delta_c = \infty$, $Flag = 1$

while *Flag* **do**

S3. $[\Delta^*_j, (j^*, b^*_j)] = \underset{j\in\mathcal{A}_t\setminus\mathcal{S}_t,\, b_j\in\mathcal{B}_j}{\arg\min} \widetilde{\Delta}^p_t(\mathcal{S}_t \cup \{j\}, b_j)$

if $\Delta^*_j < \Delta_c$ **then**

$\mathcal{S}_t \leftarrow \mathcal{S}_t \cup \{j^*\}$; $b_{j^*,t} = b^*_j$;

$\Delta_c = \Delta^*_j$;

$R^u_{j^*,t}$ as in (42); $f^l_{j^*,t}$ as in (44); f^r_t as in (46);

else

$Flag = 0$

end

end

S4. Update the federated learning algorithm as in (9);

S5. Update the virtual queues Z_t, Q_t and Y_t as in (27), (28) and (29), respectively;

end

end

learning performance and convergence rate metrics (e.g., (13) and (15)). Indeed, some convex learning tasks admit closed form expressions for different learning metrics, allowing us to use Algorithm 1. However, in several other cases (e.g., non-convex learning tasks such as deep neural network training), we do not have knowledge of such performance metrics expressions. Therefore, in this section, we extend the previous strategy incorporating an online mechanism that estimates the learning performance and the convergence rate in a fully data-driven fashion, in order to drive the dynamic resource allocation based on Lyapunov optimization. One of the nicest features of the proposed data-driven approach is that it does not necessarily rely on SGD recursions as in (9), but it works also with more sophisticated gradient-based algorithms such as, e.g., Adam, Adagrad, SGD with momentum, etc. Now, let us assume that the agents collect and process batches of data of size B_t at time-slot *t*y. For simplicity, we assume that the batch size B_t is the same for all devices, and that can be selected from a set $\mathcal{C}$ of discrete values at each time slot t. Then, assuming that N^l_i in (18) is the number of CPU cycles to compute the local gradient from one data unit, if we have batches of B_t data, the local processing time in (18) will be simply multiplied by a factor B_t. Furthermore, considering a general gradient-based algorithm (e.g., Adam), the remote processing time is obtained as in (20) simply generalizing the meaning of the constant *O*y that, starting from the received local gradient, represents the number of CPU cycles necessary to perform the single step of the gradient-based algorithm for each device. Of course, the overall remote processing time is still proportional to the number of transmitting devices. To estimate online the learning performance, 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 performance and send it (one scalar) to the server for the computation of the overall learning performance. Then, two task-dependent functions $\widehat{G}_t$ and $\widehat{\alpha}_t$ are introduced to measure online the learning performance G_t and convergence rate α_t, respectively. As an example, let us consider a classification task, whose validation (or batch) accuracy and its moving average with length $2K$ can be used to estimate learning performance and convergence rate as:

$$\widehat{G}_t = \frac{1}{|\mathcal{T}|}\sum\nolimits_{y\in\mathcal{T}} \mathbb{I}(\widehat{y}_t = y), \tag{47}$$

$$\widehat{\alpha}_t = \frac{1}{K}\sum\nolimits_{\tau=t-K}^{K-1}\Big(\widehat{G}_\tau - \widehat{G}_{\tau-1}\Big), \tag{48}$$

where $\widehat{y}_t$ is the prediction for data unit *y*y at time-slot *t*.[1] Clearly, different metrics can be used based on the task and its complexity, e.g., the ratio of gradients norms at successive time-slots can be exploited for tasks whose learning accuracy/error could be difficult or inefficient to estimate [23]. Then, we propose to exploit the performance estimates in (47)-(48) to drive the Lyapunov-based resource allocation. In particular, we introduce two new virtual queues updates, which are specific of this data-driven approach. The first virtual queue reads as:

$$\widehat{Q}_{t+1} = \max\Big\{0, \widehat{Q}_t + \epsilon_q\Big(\overline{\mathrm{G}} - \widehat{G}_t\Big)\Big\}, \tag{49}$$

and has the goal to drive the estimated learning performance $\widehat{G}_t$ above the target $\overline{\mathrm{G}}$. The second queue aims at controlling the convergence rate of the FL algorithm, and is updated as:

$$\widehat{Y}_{t+1} = \Big[\widehat{Y}_t + \epsilon_{y,t}\left(\widehat{\alpha}_t - \overline{\alpha}\right)\Big]\cdot\mathbb{I}\Big(\widehat{G}_t \leq \overline{\mathrm{G}}\Big), \tag{50}$$

where $\epsilon_{y,t}$ is an adaptive (i.e., time-dependent) step-size. The queue evolution defined in (50) 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. Thus, when $\widehat{G}_t \geq \overline{\mathrm{G}}$ (i.e., the estimated learning performance is greater than the target), the queue Y_t is set to zero and has no more impact on the convergence rate and the resource allocation. Moreover, the dynamic step-size $\epsilon_{y,t}$ is chosen to adapt the update speed of the queue Y_t depending on the distance of $\widehat{G}_t$ from the target $\overline{\mathrm{G}}$. A possible choice is $\epsilon_{y,t} = \epsilon_y|\overline{\mathrm{G}} - \widehat{G}_t|$. The rationale avoids the queue to unnecessarily impact the resource allocation when the learning performance is approaching $\overline{\mathrm{G}}$, because the target

[1]Note that, if we use the accuracy metric in (47), the long term constraints (b) in (26) must have opposite sign, i.e., the average performance must be greater than or equal to a target threshold $\overline{\mathrm{G}}$.

convergence rate $\overline{\alpha}_t$ is no more achievable at that point. At the same time, non-stationary behaviors can be detected observing a sharp deterioration in $\widehat{G}_t$, which reactivates the virtual queue Y_t, thus boosting again the learning process with the desired convergence rate. Moreover, an adaptive step-size $\epsilon_{z,t}$ can also be exploited also for the latency queue in (27) to speed up the adaptability of the method.

Using $\widehat{G}_t$ and $\widehat{\alpha}_t$ is useful for the virtual queues update in (49) and (50), but they are not explicitly (i.e., mathematically) related to the number of quantization bits and to the batch size, which must be optimized and adapted to drive the learning performance and the convergence rate. Thus, the control action might still not be easily implementable, due to the lack of closed-form expressions for the performance metrics. One possible solution to this issue builds on the following assumptions [10], which are consistently verified both from a theoretical and a numerical point of view (practical examples follow in Section V).

Assumption 3: G_t is a monotone non-decreasing function of the quantization bits $\{b_{i,t}\}_{i\in\mathcal{S}_t}$.

Assumption 4: α_t is a monotone non-decreasing function of the quantization bits $\{b_{i,t}\}_{i\in\mathcal{S}_t}$ and of the batch size B_t.

Assumption 3 hinges on the fact that a finer representation of the data generally leads to better learning performance. At the same time, Assumption 4 exploits the fact that, increasing the batch size and the number of quantization bits, the (stochastic) gradient estimates in (2) get better, thus improving the overall convergence rate. Then, under Assumption 3 and Assumption 4, we propose to exploit two surrogate functions, say $\widetilde{G}_t$ and $\widetilde{\alpha}_t$, which respectively approximate the non-decreasing behavior of G_t and α_t with respect to the quantization bits $\{b_{i,t}\}_{i\in\mathcal{S}_t}$ and the batch size B_t. Of course, there are several possible surrogates that we can exploit, but the best choice depends on the specific performance metric that we need to approximate (e.g., classification accuracy, mean-squared error, etc.). Examples will be given in Section V-B, for the case of deep neural network training. The rationale underlying the choice of the surrogates comes again from the concept of *C*yy-additive approximation [43] of the drift-plus-penalty method in (33), which makes possible to use inexact updates of the algorithm at each iteration, provided that the approximation error can be bounded within a finite error *C*. Then, at a given time slot t, exploiting (49), (50), and the surrogate functions $\widetilde{G}_t$ and $\widetilde{\alpha}_t$ in (33), we solve the following deterministic problem:

$$\min_{\mathbf{\Gamma}_t\in\mathcal{Z}_t} \quad Z_t\widetilde{L}_t - \widehat{Q}_t\widetilde{G}_t - \widehat{Y}_t\widetilde{\alpha}_t + V\cdot p_t^{\text{tot}} \tag{51}$$

where $\mathbf{\Gamma}_t = [\{b_{i,t}\}_{i\in\mathcal{S}_t}, \{R^u_{i,t}\}_{i\in\mathcal{S}_t}, \{f^l_{i,t}\}_{i\in\mathcal{S}_t}, f^r_t, B_t]$, and $\mathcal{Z}_t = \mathcal{X}_t \cup \{B_t \in \mathcal{C}\}$, with $\mathcal{C}$ denoting the discrete set of possible choices for the batch size B_t. We solve problem (51) as in the previous case, slightly modifying Algorithm 1, as illustrated in Algorithm 2. Essentially, Algorithm 2 adds a further selection step for the batch size B_t to the greedy procedure of Algorithm 1. This is done with a small additive complexity, since the number of selectable batch sizes is assumed to be small (e.g., 3 or 4 possibilities). The main steps of the proposed data-driven approach are the same as in Algorithm 1, with the difference that the virtual queues Q_t and Y_t are replaced by $\widehat{Q}_t$ and $\widehat{Y}_t$ in (49)-(50). This data-driven strategy will be numerically assessed in Section V.

Algorithm 2: Data-Driven Dynamic Resource Optimization for Adaptive Federated Learning

for $t \geq 0$ **do**
- **S1.** Observe $Z_t, \widehat{Q}_t, \widehat{Y}_t, \{h^u_{i,t}\}_{i=1}^N$ and compute $\mathcal{A}_t$ in (37).
- **if** $\mathcal{A}_t = \emptyset$ **then**
  - **continue**
- **else**
  - **S2.** Find $\mathcal{S}_t$, the quantization bits $\{b_{i,t}\}_{i\in\mathcal{S}_t}$, the batch size B_t, and the edge resources through the greedy procedure:
  - Set $\mathcal{S}_t = \emptyset$, $\{b_{i,t}\}_{i=1}^N = 0$, $\Delta_c = \infty$, $Flag = 1$
  - **for** $B \in \mathcal{C}$ **do**
    - **while** *Flag* **do**
      - **S2.** $[\Delta^*_j, (j^*, b^*_j)] = \arg\min_{j\in\mathcal{A}_t\setminus\mathcal{S}_t, b_j\in\mathcal{B}_j} \widetilde{\Delta}^p_t(\mathcal{S}_t\cup\{j\}, b_j, B)$
      - **if** $\Delta^*_j < \Delta_c$ **then**
        - $\mathcal{S}_t \leftarrow \mathcal{S}_t \cup \{j^*\}$; $b_{j^*,t} = b^*_j$;
        - $\Delta_c = \Delta^*_j$;
        - $R^u_{j^*,t}$ as in (42); $f^l_{j^*,t}$ as in (44); f^r_t as in (46);
        - $B_t = B$
      - **else**
        - $Flag = 0$
      - **end**
    - **end**
  - **end**
  - **S3.** Update the federated learning algorithm as in (9), or according to a tailored gradient-based optimizer;
  - **S4.** Update the virtual queues Z_t, $\widehat{Q}_t$ and $\widehat{Y}_t$ as in (27), (49) and (50), respectively;
- **end**

end

Remark 2: Interestingly, Algorithms 1 and 2 implement a double-step struggler mitigation at each time-slot, which selectively avoids that worst-case devices hinder the performance of the proposed strategy. First, there is a radio admission control step defined in (37), which selects the set $\mathcal{A}_t$ of agents that can transmit with a minimum rate $R^{\min}$, thus discarding the agents experiencing bad wireless channel conditions (and, consequently, high communication latency). Then, starting from the set $\mathcal{A}_t$ of potential transmitters, Algorithms 1 and 2 choose the set $\mathcal{S}_t \subseteq \mathcal{A}_t$ of transmitting agents in order to minimize the per-slot optimization problems in (33) and (51), respectively. Since the objectives of (33) and (51) encompass jointly power, latency, and learning performance of the FL task, this second struggler mitigation step selects the subset $\mathcal{S}_t$ in order

to strike the best trade-off between these three fundamental aspects of the problem.

V. Numerical Results

In this section, we assess the performance of the proposed method, considering both model-based and data-driven scenarios. In particular, we will exploit the model-based approach for a least-mean squares regression task (in Section V-A), and the data-driven approach for a classification task aimed at training a deep convolutional neural network (in Section V-B). We consider a scenario with $N = 9$ devices and one AP equipped with an edge server, as illustrated in Fig. 1. We set the radio and computation parameters as follows: $N_0 = -174$ dBm/Hz, $A = 25$, $f_r^{\max} = 3.3$ GHz, $f_i^{\max} = 2.5$ GHz, $R_{i,t}^{\max} = B_{i,t}^u \log_2(1 + h_{i,t}^u P_i^{\max}/(N_0 B_{i,t}^u))$ where $P_i^{\max} = 15$ mW for all i; $B_{i,u}^t$ is assigned equally splitting the overall bandwidth among the devices transmitting at time t. Moreover, $\mathcal{B}_i = [2, 4, 8]$, and $R_{i,t}^d = R_{i,t}^{\max}$ for all i, and t. The channels are generated using the ABG model [53], with a carrier frequency of 23 GHz and adding a Rayleigh fading with unit variance. The LMS results are obtained with MATLAB, using a PC with an Intel Core i7-7700HQ CPU at a frequency of 2.80 GHz. The data-driven results are obtained using Python and the JAX framework, exploiting an NVIDIA Tesla K80 GPU.

A. Federated Least-Mean Squares

For this learning task, the input data $\boldsymbol{x}_{i,t} \in \mathbb{R}^{20}$ is related to the corresponding output via a linear model $y_{i,t} = \boldsymbol{x}_{i,t}^T \mathbf{w}^{\mathrm{o}} + v_{i,t}$, at each time instant t. In this context, the SGD algorithm in (2) boils down into a federated LMS adaptive algorithm aimed at learning (and tracking over time) the vector $\mathbf{w}^{\mathrm{o}}$ [42]. The devices locations are chosen at random such that the distance of each device from the AP is sampled from a uniform distribution in the interval [70, 130], for all $i = 1, \ldots, N$. We assume that input data $\boldsymbol{x}_{i,t}$ are zero-mean random vectors with covariance matrix $\sigma_{x,i}^2 \mathbf{I}_{20}$ with $\sigma_{x,i}^2 = 1$ for all i. The observation noise $v_{i,t}$ is Gaussian, zero-mean with variance $\sigma_{v,i}^2$ uniformly selected in the interval $[0, 2 \times 10^{-3}]$, for all $i = 1, \ldots, N$, independent from the data and among devices. Also, the overall bandwidth is 100 KHz, $O = 2 \cdot 10^3$, $N_i^l = 5 \cdot 10^6$. The step-size μ is set to 0.015. The learning performance is measured in terms of MSD as in (13), and the convergence rate is given by (15). As a first result, in Fig. 2, we illustrate the learning curve of the FL algorithm in (2), obtained for different values of learning performance constraints $\overline{\mathrm{G}} = \overline{\mathrm{MSD}}$, while fixing the convergence rate to $\overline{\alpha} = 0.99$ and the latency constraint $\overline{\mathrm{L}} = 20$ ms. The results are averaged over 50 independent simulations, setting empirically ϵ_z, ϵ_q, ϵ_y to obtain the fastest convergence. From Fig. 2, we can notice how the proposed optimization method is able to guarantee the prescribed performance in terms of convergence rate $\overline{\alpha}$ and steady-state accuracy $\overline{\mathrm{MSD}}$. Then, in Fig. 3, we show the histogram of quantization bits usage for different values of $\overline{\mathrm{MSD}}$, fixing $\overline{\alpha} = 0.99$ and $\overline{\mathrm{L}} = 20$ ms. From Fig. 3, we notice how the method requires on average more quantization bits to obtain a stricter requirement

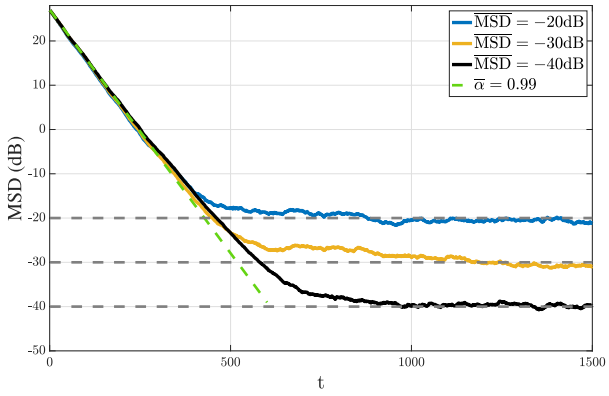


Fig. 2. MSD vs *t*yy, for different values of $\overline{\mathrm{MSD}}$.

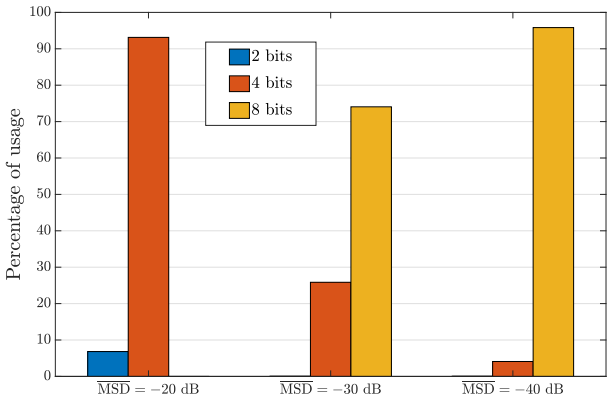
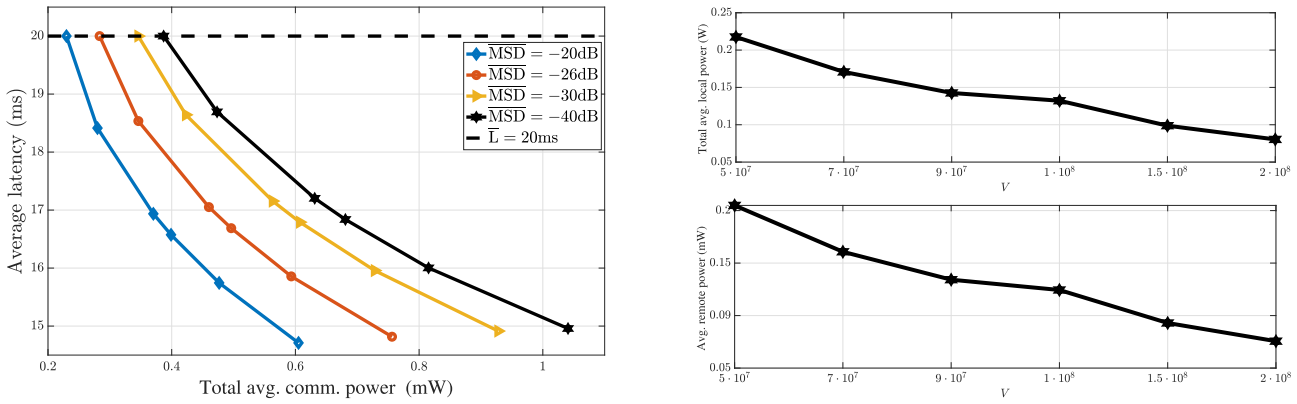


Fig. 3. Quantization bits, for different $\overline{\mathrm{MSD}}$.

on learning performance, due to the finer required representation of transmitted data. Finally, in Fig. 4 (a) we illustrate the trade-off curve between average latency and TX power consumption (i.e., the sum of powers transmitted by all users) achieved by the proposed method, considering different values of $\overline{\mathrm{MSD}}$ and fixed $\overline{\alpha} = 0.99$. Each point in the curves of Fig. 4 (a) represents a different value of V, whose magnitude grows from right to left. From Fig. 4 (a), increasing V, the method reduces the transmission power up to a limit value (i.e., the optimum) that still enables to guarantee the target latency constraint. As expected, the trade-off gets worse imposing a stricter requirement on learning performance, due to the larger power (and number of bits) necessary to obtain the target performance. Also, from a computation point of view, in Figs. 4 (b) we illustrate the average remote and local processing power consumption vs V, fixing $\overline{\mathrm{MSD}} = -40$ dB, $\overline{\alpha} = 0.99$ and $\overline{\mathrm{L}} = 20$ ms. As expected, the proposed method is able to reduce all the single contributions of the overall power consumption as V increases.

B. Federated Deep Neural Network Training

In this section, we consider a learning task aimed at training a classifier based on deep convolutional neural networks (CNN). We exploit a CNN made of four convolutional layers with 32, 32, 10 and 10 filters, respectively, with final flatten and dense layers; SAME padding, ReLu non-linearities and Batch Normalization are applied after each convolutional



(a) (b)

Fig. 4. (a) Latency vs power consumption, (b)-(c) Power consumption vs V for $\overline{\text{MSD}} = -40$ dB.

layer, and a final Softmax non-linearity is applied after the flatten and dense layers. The filters dimensions are 5×5, 5×5, and 3×3 and 3×3, respectively. To train the CNN, we use the MNIST dataset [54], which is made of 28×28 grayscale images of handwritten digits divided in 10 classes. The training data is composed of 6×10^4 examples, while the test set is made of 10^4 elements. The loss is the well-known cross-entropy, and the model is trained using a federated ADAM optimizer [55]. The devices locations are random and the distance of each device from the AP is sampled from a uniform distribution in the interval [20, 80], for all $i = 1, \ldots, N$. Also, the overall bandwidth is 100 MHz, $O = 10^6$, and $N_i^l = 10^7$. The ADAM step-size is set to 0.008, with forgetting factors $\beta_1 = 0.9$, and $\beta_2 = 0.99$. For this experiment, we use the performance estimate $\widehat{G}_t$ in (47), and $\widehat{\alpha}_t$ in (48) with $K = 10$. Also, as a surrogate function for the accuracy metric, we exploit:

$$\widetilde{G}_t = \sum\nolimits_{i \in \mathcal{S}_t} \sigma(b_{i,t} - \text{Median}\{\mathcal{B}_i\}), \tag{52}$$

where $\sigma(\cdot)$ is the logistic sigmoid function, and Median$\{\cdot\}$ represents the median value. Clearly, (52) satisfies Assumption 3. Regarding the convergence rate, we use instead the surrogate $\widetilde{\alpha}_t = B_t \sum_{i \in \mathcal{S}_t} b_{i,t}$, which of course satisfies Assumption 4. Moreover, $\mathcal{C} = [1, 3, 7, 14]$ for Fig. 5 and $\mathcal{C} = [1, 3, 7]$ for the others. As a first result, in Fig. 5, we illustrate the temporal behavior of the estimated accuracy of the FL algorithm, obtained for different values of the convergence rate $\overline{\alpha}$, while fixing the accuracy to $\overline{\text{G}} = 0.8$ and the latency constraint $\overline{\text{L}} = 50$ ms. As we can notice form Fig. 5, the proposed data-driven method is able strike the desired learning performance, while controlling the convergence rate. Similarly, in Fig. 6 (a), we illustrate the temporal behavior of the estimated accuracy, obtained for different values of the performance constraint $\overline{\text{G}}$, while fixing the convergence rate to $\overline{\alpha} = 0.2$ and the latency constraint $\overline{\text{L}} = 50$ ms. Then, in Fig. 6 (b) we show the temporal evolution of the overall latency and the overall uplink transmission power consumption, respectively, corresponding to Fig. 6 for $\overline{\text{G}} = 0.8$ and $\overline{\text{G}} = 0.9$. As we can notice from Fig. 6, the proposed method keeps the latency around the requirement $\overline{\text{L}}$, while driving the

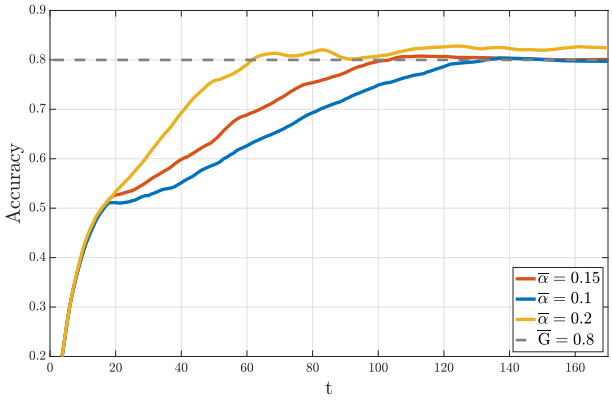
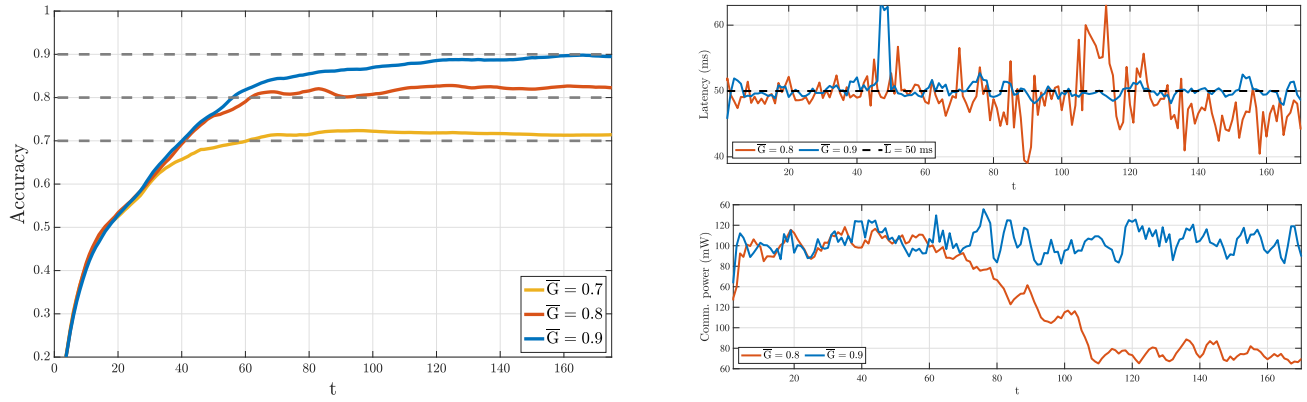


Fig. 5. Accuracy vs t, for different values of $\overline{\alpha}$.

accuracy on the target value $\overline{\text{G}}$ during the steady state phase. Interestingly, from Fig. 6 (c), we notice how a significant power saving can be achieved at steady-state if the accuracy requirement is not very strict (i.e., for $\overline{\text{G}} = 0.8$), thanks to the impact of the adaptive step-size in (50). Furthermore, the results empirically confirm the choice of the adopted surrogate functions, and the consistence of Assumptions 3 and 4.

Comparisons: Even though there are several works on resource allocation for FL, our problem formulation, jointly encompassing communication, computation and learning aspects of FL in a dynamic and adaptive fashion is novel, and does not come from a straightforward modification of existing results. Thus, it is extremely difficult to provide fair comparisons with other techniques available in the literature. However, we decided to assess the advantages of our joint strategy by comparing it with (sub-)procedures involving the optimization only of single aspects. We consider the following strategies for comparison: i) Equal Rate Policy (referred to as *Equal Rate*): All the agents always transmit with a fixed number of quantization bits (to match a certain learning accuracy), the remote and local frequencies are fixed, and the uplink rate is equally adapted for all the agents to match the latency constraint; ii) Fixed Scheduling & Quantization Bits Policy (referred to as *Fixed S&B*): All the agents always



(a) (b)

Fig. 6. (a) Accuracy vs t, (b) Latency vs t and Power consumption vs t, for different $\overline{G}$.

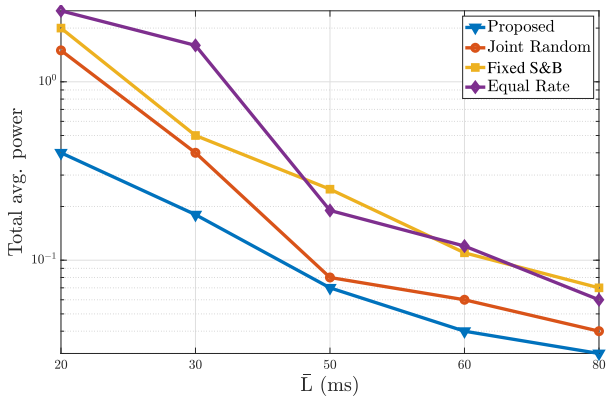


Fig. 7. Power consumption vs latency.

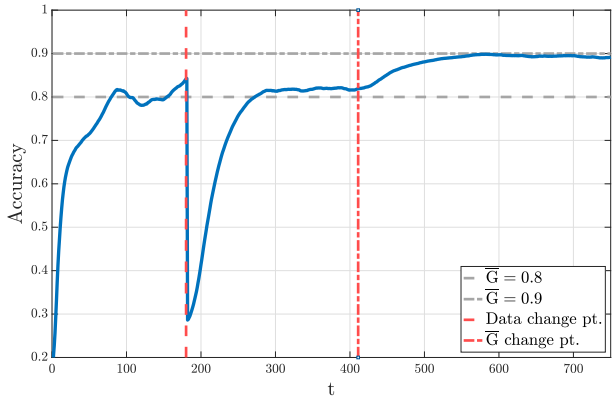


Fig. 8. Accuracy vs t in a non-stationary.

transmit with a fixed number of quantization bits (to match a certain learning accuracy), whereas remote frequency, local frequencies and uplink rates are optimized via Lyapunov Optimization. iii) Joint Optimization with Random Scheduling and Quantization Bits (referred to as *Random Joint*): It is our joint procedure, but the scheduling and quantization bits are not assigned via the proposed greedy method, but rather with a random search of comparable complexity, meaning that multiple random realizations of the variables are checked to select the best one. In Fig. 7, we illustrate the trade-off curve between average total power consumption and average latency $\overline{L}$ for the aforementioned strategies, referring to our procedure as *Proposed*. The values are obtained by fixing the accuracy threshold $\bar{G} = 0.8$, the frequencies for *Equal Rate* strategy to the average frequencies of our procedure, the quantization bits to six for *Equal Rate* and *Fixed S&B* strategies (to tightly match the accuracy constraint). From Fig. 7, the proposed method results in a better trade-off with respect to the other strategies, i.e., in a sensible power saving for any given delay. Moreover, we empirically observed that our strategy is the only one effectively able to control also the convergence rate. Although the proposed comparisons do not (and cannot) refer to specific other works, they follow the optimization approach (in terms of optimization variables) of other works, e.g., [39] (they optimize only scheduling and power allocation), [36] (they optimize quantization schemes), [22] (they optimize scheduling, power allocation and RB-OFDMA allocation).

Adaptation in non-stationary conditions: Finally, in Fig. 8 we illustrate the temporal behavior of the estimated accuracy in a non-stationary scenario, in order to highlight the adaptation capabilities of the proposed method. In particular, the MNIST dataset is divided into two sub-dataset of 5 classes each; then, the architecture is trained for the first 180 time-slot with one of the two sub-datasets and for the remaining time-slots with the other one (the last dense layer is obviously reduced to a 5 dimensional output). Equivalently, this introduces a non-stationary behavior of the data distribution. Then, at time slot 401, we change the accuracy requirement from $\overline{G} = 0.8$ to $\overline{G} = 0.9$, introducing a further level of non-stationarity. The results are averaged over 10 independent simulations. From Fig. 8, we can notice that our dynamic strategy is able to react promptly to both changes in the data distribution and in the accuracy requirement, exhibiting powerful learning and adaptation capabilities in a fully data-driven fashion.

VI. Conclusion

In this paper, we have proposed a dynamic resource allocation strategy enabling adaptive federated learning at the wireless network edge. The strategy dynamically minimizes the power expenditure of the system, while guaranteeing target

learning performance and latency constraints. The proposed method builds on stochastic Lyapunov optimization, which leads to low-complexity procedures for the resource allocation at each time slot, without requiring a-priori knowledge of wireless channel statistics. The approach is valid both for a model-based approach, where performance metrics can be evaluated in closed-form, or for a data-driven approach, where performance are estimated online from streaming data. Several numerical results assess the performance of the proposed strategy over both synthetic and real data. Future research directions include model-based approaches for the non-convex learning scenario, where theoretical expressions for the convergence rates to (local) optimality can be used to control the performance of FL.

APPENDIX

Let us present the derivations leading to the upper bound in (32). In particular, considering Z_t defined in (27) and defining $\Delta_Z = \frac{Z_{t+1}^2 - Z_t^2}{2}$, we have [43]:

$$\Delta_Z \leq \frac{\left(L_t - \bar{L}\right)^2}{2} + Z_t(L_t - \bar{L}). \tag{53}$$

From (26), since we assume a minimum uplink data rate $R_i^{\min}$ and a minimum local CPU frequency $f_i^{\min}$ for every device $i \in \mathcal{S}_t$, and a minimum server CPU clock frequency $f^{r,\min}$, the term L_t in (53) is bounded for all t by a finite constant $L^{\mathrm{SGD,max}}$, i.e.,

$$\Delta_Z \leq \frac{(L^{\mathrm{SGD,max}} - \bar{L})^2}{2} + Z_t\big(L_t - \bar{L}\big). \tag{54}$$

Applying the same arguments to Q_t defined in (28), and exploiting the upper-bound $G_t \leq G^{\max}$ (which holds for any suitable performance metric), we obtain:

$$\Delta_Q \leq \frac{(G^{\max} - \overline{\mathrm{G}})^2}{2} + Q_t(G_t - \overline{\mathrm{G}}). \tag{55}$$

This last condition holds due to (16) and the fact that we impose $\sum_{i=1}^{N} a_{i,t} \geq 1$ (cf. (26)). Finally, let us consider the virtual queue Y_t defined in (29). Although Y_t presents a different evolution from the other virtual queues, we can still apply the same arguments and, exploiting the upper-bound $\alpha_t \leq \alpha^{\max}$ (which holds for any metric of convergence rate), we obtain:

$$\Delta_Y \leq \frac{(\alpha_t^{\max} - \bar{\alpha})^2}{2} + Y_t(\alpha_t - \bar{\alpha}). \tag{56}$$

Finally, plugging (54), (55) and (56) into (31), we derive the upper bound in (32), i.e.,

$$\Delta_t^p \leq \zeta + \mathbb{E}\Big\{ Z_t\big(L_t - \bar{L}\big) + Q_t\big(G_t - \overline{G}\big) + Y_t(\alpha_t - \bar{\alpha}) + V p_t^{\mathrm{tot}} \Big| \mathbf{\Phi}_t \Big\},$$

where ζ is a finite positive constant that reads as

$$\zeta = \frac{\left(L^{\mathrm{SGD,max}} - \bar{L}\right)^2}{2} + \frac{\left(G^{\max} - \overline{\mathrm{G}}\right)^2}{2} + \frac{(\alpha^{\max} - \bar{\alpha})^2}{2} \tag{57}$$

REFERENCES

[1] S. Ahmadi, *5G NR: Architecture, Technology, Implementation, and Operation of 3GPP New Radio Standards*. London, U.K.: Elsevier Sci., 2019.

[2] E. C. Strinati *et al.*, "6G: The next frontier: From holographic messaging to artificial intelligence using subterahertz and visible light communication," *IEEE Veh. Technol. Mag.*, vol. 14, no. 3, pp. 42–50, Sep. 2019.

[3] A. Ndikumana *et al.*, "Joint communication, computation, caching, and control in big data multi-access edge computing," *IEEE Trans. Mobile Comput.*, vol. 19, no. 6, pp. 1359–1374, Jun. 2020.

[4] S. Barbarossa, S. Sardellitti, and P. Di Lorenzo, "Communicating while computing: Distributed mobile cloud computing over 5G heterogeneous networks," *IEEE Signal Process. Mag.*, vol. 31, no. 6, pp. 45–55, Nov. 2014.

[5] E. C. Strinati *et al.*, "Wireless environment as a service enabled by reconfigurable intelligent surfaces: The RISE-6G perspective," in *Proc. EUCNC 6G Summit*, Porto, Portugal, Jun. 2021, pp. 562–567.

[6] M. Polese, R. Jana, V. Kounev, K. Zhang, S. Deb, and M. Zorzi, "Machine learning at the edge: A data-driven architecture with applications to 5G cellular networks," *IEEE Trans. Mobile Comput.*, vol. 20, no. 12, pp. 3367–3382, Dec. 2021.

[7] J. Park, S. Samarakoon, M. Bennis, and M. Debbah, "Wireless network intelligence at the edge," *Proc. IEEE*, vol. 107, no. 11, pp. 2204–2239, Nov. 2019.

[8] J. Chen and X. Ran, "Deep learning with edge computing: A review," *Proc. IEEE*, vol. 107, no. 8, pp. 1655–1674, Aug. 2019.

[9] X. Wang, Y. Han, V. C. M. Leung, D. Niyato, X. Yan, and X. Chen, "Convergence of edge computing and deep learning: A comprehensive survey," *IEEE Commun. Surveys Tuts.*, vol. 22, no. 2, pp. 869–904, 2nd Quart., 2020.

[10] 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.

[11] 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.

[12] J. Konečnỳ, H. B. McMahan, D. Ramage, and P. Richtárik, "Federated optimization: Distributed machine learning for on-device intelligence," 2016, *arXiv:1610.02527*.

[13] P. H. Jin, Q. Yuan, F. Iandola, and K. Keutzer, "How to scale distributed deep learning?" 2016, *arXiv:1611.04581*.

[14] T. Li, A. K. Sahu, M. Zaheer, M. Sanjabi, A. Talwalkar, and V. Smith, "Federated optimization in heterogeneous networks," 2018, *arXiv:1812.06127*.

[15] 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. AISTATS*, Lauderdale, FL, USA, Apr. 2017, pp. 1273–1282.

[16] M. S. H. Abad, E. Ozfatura, D. Gunduz, and O. Ercetin, "Hierarchical federated learning across heterogeneous cellular networks," in *Proc. IEEE ICASSP*, Barcellona, Spain, May 2020, pp. 8866–8870.

[17] J. Konečnỳ, B. McMahan, and D. Ramage, "Federated optimization:Distributed optimization beyond the datacenter," 2015, *arXiv:1511.03575*.

[18] S. Samarakoon, M. Bennis, W. Saad, and M. Debbah, "Distributed federated learning for ultra-reliable low-latency vehicular communications," *IEEE Trans. Commun.*, vol. 68, no. 2, pp. 1146–1159, Feb. 2020.

[19] S. Ha, J. Zhang, O. Simeone, and J. Kang, "Coded federated computing in wireless networks with straggling devices and imperfect CSI," in *Proc. IEEE ISIT*, Paris, France, Jul. 2019, pp. 2649–2653.

[20] S. Wang *et al.*, "Adaptive federated learning in resource constrained edge computing systems," *IEEE J. Sel. Areas Commun.*, vol. 37, no. 6, pp. 1205–1221, Jun. 2019.

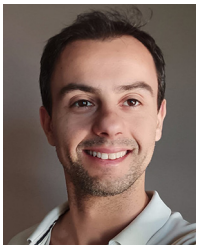
[21] Q. Zeng, K. Du, Y. Huang, and K. K. Leung, "Energy-efficient radio resource allocation for federated edge learning," in *Proc. IEEE ICC Workshops*, Dublin, Ireland, Jun. 2020, pp. 1–6.

[22] M. Chen, Z. Yang, W. Saad, C. Yin, H. V. Poor, and S. Cui, "A joint learning and communications framework for federated learning over wireless networks," *IEEE Trans. Wireless Commun.*, vol. 20, no. 1, pp. 269–283, Jan. 2021.

[23] N. H. Tran, W. Bao, A. Zomaya, M. N. H. Nguyen, and C. S. Hong, "Federated learning over wireless networks: Optimization model design and analysis," in *Proc. IEEE INFOCOM*, Paris, France, Sep. 2019, pp. 1387–1395.

[24] H. H. Yang, Z. Liu, T. Q. S. Quek, and H. V. Poor, "Scheduling policies for federated learning in wireless networks," *IEEE Trans. Commun.*, vol. 68, no. 1, pp. 317–333, Jan. 2020.
[25] Z. Ma, Y. Xu, H. Xu, Z. Meng, L. Huang, and Y. Xue, "Adaptive batch size for federated learning in resource-constrained edge computing," *IEEE Trans. Mobile Comput.*, early access, Apr. 23, 2021, doi: 10.1109/TMC.2021.3075291.
[26] L. U. Khan *et al.*, "Federated learning for edge networks: Resource optimization and incentive mechanism," *IEEE Commun. Mag.*, vol. 58, no. 10, pp. 88–93, Oct. 2020.
[27] Z. Yang, M. Chen, W. Saad, C. S. Hong, and M. Shikh-Bahaei, "Energy efficient federated learning over wireless communication networks," *IEEE Trans. Wireless Commun.*, vol. 20, no. 3, pp. 1935–1949, Mar. 2021.
[28] 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.
[29] J. Mills, G. Hu, and J. Min, "Communication-efficient federated learning for wireless edge intelligence in IoT," *IEEE Internet Things J.*, vol. 7, no. 7, pp. 5986–5994, Jul. 2020.
[30] Y. Jiang *et al.*, "Model pruning enables efficient federated learning on edge devices," 2020, *arXiv:1909.12326*.
[31] X. Mo and J. Xu, "Energy-efficient federated edge learning with joint communication and computation design," *J. Commun. Inf. Netw.*, vol. 6, no. 2, pp. 110–124, Jun. 2021.
[32] 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. Wireless Commun.*, vol. 20, no. 1, pp. 453–467, Jan. 2021.
[33] M. Chen, H. V. Poor, W. Saad, and S. Cui, "Convergence time optimization for federated learning over wireless networks," *IEEE Trans. Wireless Commun.*, vol. 20, no. 4, pp. 2457–2471, Apr. 2021.
[34] B. Luo, X. Li, S. Wang, J. Huang, and L. Tassiulas, "Cost-effective federated learning in mobile edge networks," *IEEE J. Sel. Areas Commun.*, vol. 39, no. 12, pp. 3606–3621, Dec. 2021.
[35] Z. Zhao, J. Xia, L. Fan, X. Lei, G. K. Karagiannidis, and A. Nallanathan, "System optimization of federated learning networks with a constrained latency," *IEEE Trans. Veh. Technol.*, vol. 71, no. 1, pp. 1095–1100, Jan. 2022.
[36] 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, Dec. 2020.
[37] S. Chen, C. Shen, L. Zhang, and Y. Tang, "Dynamic aggregation for heterogeneous quantization in federated learning," *IEEE Trans. Wireless Commun.*, vol. 20, no. 10, pp. 6804–6819, Oct. 2021.
[38] Y. Wang, Y. Xu, Q. Shi, and T.-H. Chang, "Quantized federated learning under transmission delay and outage constraints," *IEEE J. Sel. Areas Commun.*, vol. 40, no. 1, pp. 323–341, Jan. 2022.
[39] Y. Sun, S. Zhou, and D. Gündüz, "Energy-aware analog aggregation for federated learning with redundant data," in *Proc. IEEE ICC*, Jun. 2020, pp. 1–7.
[40] S. Zhou, Z. Yang, L. Pu, and S. Yu, "CEFL: Online admission control, data scheduling, and accuracy tuning for cost-efficient federated learning across edge nodes," *IEEE Internet Things J.*, vol. 7, no. 10, pp. 9341–9356, Oct. 2020.
[41] T. Huang, W. Lin, W. Wu, L. He, K. Li, and A. Y. Zomaya, "An efficiency-boosting client selection scheme for federated learning with fairness guarantee," *IEEE Trans. Parallel Distrib. Syst.*, vol. 32, no. 7, pp. 1552–1564, Jul. 2021.
[42] A. H. Sayed, "Adaptation, learning, and optimization over networks," *Found. Trends Mach. Learn.*, vol. 7, pp. 311–801, Jul. 2014.
[43] M. J. Neely, *Stochastic Network Optimization with Application to Communication and Queueing Systems*. San Rafael, CA, USA: Morgan Claypool, 2010.
[44] P. Di Lorenzo, C. Battiloro, M. Merluzzi, and S. Barbarossa, "Dynamic resource optimization for adaptive federated learning at the wireless network edge," in *Proc. IEEE ICASSP*, Toronto, ON, Canada, Jun. 2021, pp. 4910–4914.
[45] N. Agarwal, A. T. Suresh, F. Yu, S. Kumar, and H. B. Mcmahan, "CpSGD: Communication-efficient and differentially-private distributed SGD," in *Proc. NIPS*, Montreal, QC, Canada, Dec. 2018, pp. 7575–7586.
[46] W.-T. Chang and R. Tandon, "Communication efficient federated learning over multiple access channels," 2020, *arXiv:2109.05411*.
[47] S. P. Lipshitz, R. A. Wannamaker, and J. Vanderkooy, "Quantization and dither: A theoretical survey," *J. Audio Eng. Soc.*, vol. 40, no. 5, pp. 355–375, May 1992.
[48] R. A. Wannamaker, S. P. Lipshitz, J. Vanderkooy, and J. N. Wright, "A theory of nonsubtractive dither," *IEEE Trans. Signal Process.*, vol. 48, no. 2, pp. 499–516, Feb. 2000.
[49] L. Schuchman, "Dither signals and their effect on quantization noise," *IEEE Trans. Commun. Technol.*, vol. TCT-12, no. 4, pp. 162–165, Dec. 1964.
[50] T. D. Burd and R. W. Brodersen, "Processor design for portable systems," *J. VLSI Signal Process. Syst. Signal Image Video Technol.*, vol. 13, nos. 2–3, pp. 203–221, Aug. 1996.
[51] M. Merluzzi, P. Di Lorenzo, S. Barbarossa, and V. Frascolla, "Dynamic computation offloading in multi-access edge computing via ultra-reliable and low-latency communications," *IEEE Trans. Signal Inf. Process. Netw.*, vol. 6, pp. 342–356, Mar. 2020.
[52] P. Brito, F. Fabião, and A. Staubyn, "Euler, Lambert, and the Lambert W-function today," *Math. Sci.*, vol. 33, pp. 127–133, Jan. 2008.
[53] G. R. MacCartney, S. Deng, S. Sun, and T. S. Rappaport, "Millimeter-wave human blockage at 73 GHz with a simple double knife-edge diffraction model and extension for directional antennas," in *Proc. IEEE VTC-Fall*, Montreal, QC, Canada, Sep. 2016, pp. 1–6.
[54] Y. Lecun, L. Bottou, Y. Bengio, and P. Haffner, "Gradient-based learning applied to document recognition," *Proc. IEEE*, vol. 86, no. 11, pp. 2278–2324, Nov. 1998.
[55] D. Kingma and J. Ba, "Adam: A method for stochastic optimization," in *Proc. ICLR*, Banff, AB, Canada, Apr. 2014, pp. 1–15.

Claudio Battiloro (Graduate Student Member, IEEE) received the bachelor's degree in computer engineering from the University of Perugia, Perugia, Italy, in 2018, and the M.Sc. degree (*cum laude* and Hons.) in data science from the Sapienza University of Rome, Rome, Italy, in 2020, where he is currently pursuing the Ph.D. degree in information and communication technologies with the Department of Information Engineering, Electronics, and Telecommunications. His research interests include topological signal processing theory and methods, geometric deep learning, edge machine learning, and distributed optimization. He has been awarded the prize for the Best M.Sc. Thesis for the year 2019/2020 called by the IEEE SPS Italian Chapter.



Paolo Di Lorenzo (Senior Member, IEEE) received the M.Sc. and Ph.D. degrees in electrical engineering from the Sapienza University of Rome, Rome, Italy, in 2008 and 2012, respectively, where he is an Associate Professor with the Department of Information Engineering, Electronics, and Telecommunications. In 2010, he held a visiting research appointment with the Department of Electrical Engineering, University of California at Los Angeles, Los Angeles, CA, USA. He has participated in the FP7 European research projects FREEDOM, on femtocell networks; SIMTISYS, on moving target detection and imaging using a constellation of satellites; and TROPIC, on communication, computation, and storage over collaborative femtocells. He is the Principal Investigator of CNIT-Sapienza Research Unit in the H2020 European Project RISE 6G. His research interests include signal processing theory and methods, wireless edge learning, distributed optimization, machine learning, and (beyond)graph signal processing. He is the recipient of the 2022 EURASIP Early Career Award. He is the recipient of three best student paper awards, respectively, at IEEE SPAWC10, EURASIP EUSIPCO11, and IEEE CAMSAP11. He is also the recipient of the 2012 GTTI (Italian National Group on Telecommunications and Information Theory) Award for the Best Ph.D. Thesis in communication engineering. He is currently an Associate Editor of the IEEE TRANSACTIONS ON SIGNAL AND INFORMATION PROCESSING OVER NETWORKS.

Mattia Merluzzi (Member, IEEE) received the M.S. degree in telecommunication engineering and the Ph.D. degree in information and communication technologies from the Sapienza University of Rome, Italy, in 2017 and 2021, respectively. He is currently a Research Engineer with CEA-Leti, Grenoble, France, where he is involved in the research team of the H2020 Project Hexa-X. He has participated in the H2020 EU/Japan Project 5G-Miedge, the H2020 EU/Taiwan Project 5G CONNI, and the MIUR funded PRIN Liquid Edge. His primary research interests are in edge computing, beyond 5G and 6G systems, stochastic optimization, and edge machine learning. He was the recipient of the 2021 GTTI (Italian National Group on Telecommunications and Information Theory) Award for the Best Ph.D. Thesis.

Sergio Barbarossa (Fellow, IEEE) received the M.S. and Ph.D. degrees in electrical engineering from the Sapienza University of Rome, Rome, Italy, where he is currently a Full Professor and a Senior Research Fellow with the Sapienza School for Advanced Studies. He has held visiting positions with the Environmental Research Institute of Michigan in 1988, the University of Virginia in 1995 and 1997, and the University of Minnesota in 1999. He has been the Scientific Coordinator of many international projects on wireless sensor networks, small cell networks, distributed mobile cloud computing, and edge computing in 5G networks. He is currently leading the national project Liquid Edge, on edge learning and he is involved in the H2020 European projects 5G-CONNI, on 5G networks for Industry 4.0, and RISE-6G on reconfigurable intelligent surfaces. His research interests include 6G networks, semantic communications, edge machine learning, and topological signal processing and learning. He received the IEEE Best Paper Awards from the IEEE Signal Processing Society in 2000, 2014, and 2020. He received the 2010 Technical Achievements Award from the European Association for Signal Processing Society. He served as an IEEE Distinguished Lecturer. He is a EURASIP Fellow.