IEEE Access
Multidisciplinary ⋮ Rapid Review ⋮ Open Access Journal

Received 1 February 2023, accepted 25 February 2023, date of publication 2 March 2023, date of current version 8 March 2023.

Digital Object Identifier 10.1109/ACCESS.2023.3251571

RESEARCH ARTICLE

A Layer Selection Optimizer for Communication-Efficient Decentralized Federated Deep Learning

LUCA BARBIERI[1], (Graduate Student Member, IEEE), STEFANO SAVAZZI[2], (Member, IEEE), AND MONICA NICOLI[3], (Senior Member, IEEE)

[1]Dipartimento di Elettronica, Informazione e Bioingegneria, Politecnico di Milano, 20133 Milan, Italy
[2]Institute of Electronics, Information and Telecommunications Engineering (IEIIT), Consiglio Nazionale delle Ricerche, 20133 Milan, Italy
[3]Dipartimento di Ingegneria Gestionale, Politecnico di Milano, 20156 Milan, Italy

Corresponding author: Luca Barbieri (luca1.barbieri@polimi.it)

This work was supported in part by the Project BASE5G funded by Regione Lombardia within the framework POR FESR 2014–2020 under Project 1155850, and in part by the CHIST-ERA EU Project RadioSense (Wireless Big-Data Augmented Smart Industry) under Grant CHIST-ERA-17-BDSI-005.

ABSTRACT Federated Learning (FL) systems orchestrate the cooperative training of a shared Machine Learning (ML) model across connected devices. Recently, decentralized FL architectures driven by consensus have been proposed to enable the devices to share and aggregate the ML model parameters via direct sidelink communications. The approach has the advantage of promoting the federation among the agents even in the absence of a server, but may require an intensive use of communication resources compared to vanilla FL methods. This paper proposes a communication-efficient design of consensus-driven FL optimized for training of Deep Neural Networks (DNNs). Devices independently select fragments of the DNN to be shared with neighbors on each training round. Selection is based on a local optimizer that trades model quality improvement with sidelink communication resource savings. The proposed technique is validated on a vehicular cooperative sensing use case characterized by challenging real-world datasets and complex DNNs typically employed in autonomous driving with up to 40 trainable layers. The impact of layer selection is analyzed under different distributed coordination configurations. The results show that it is better to prioritize the DNN layers possessing few parameters, while the selection policy should optimally balance gradient sorting and randomization. Latency, accuracy and communication tradeoffs are analyzed in detail targeting sustainable federation policies.

INDEX TERMS Machine learning over networks, federated learning, consensus, sidelink communications, beyond 5G.

I. INTRODUCTION

Distributed learning methodologies based on consensus [1], [2], [3] have emerged over the last few years for solving complex processing [4], [5], [6], [7] and decision-making tasks [8], [9] over cooperative networks. In this paradigm, interconnected agents combine local processing procedures with mutual interactions over a mesh network to learn a shared model describing the task to be fulfilled [4], [6]. Centralized learning implementations that involve energy-intensive processing at data centers or servers can be avoided by promoting nodes' self-organization via consensus methods [1], [3]. This approach is expected to bring significant advantages in terms of latency, scalability, and robustness, especially within new-generation wireless networks. 6th Generation (6G) cellular systems are in fact moving towards dedicated infrastructures [10], [11], [12] to support decentralized, device-to-device communications [13], [14] tailored for specific industry verticals, ranging from robotics [15] to autonomous driving [16], [17].

The associate editor coordinating the review of this manuscript and approving it for publication was Ahmed Almradi.

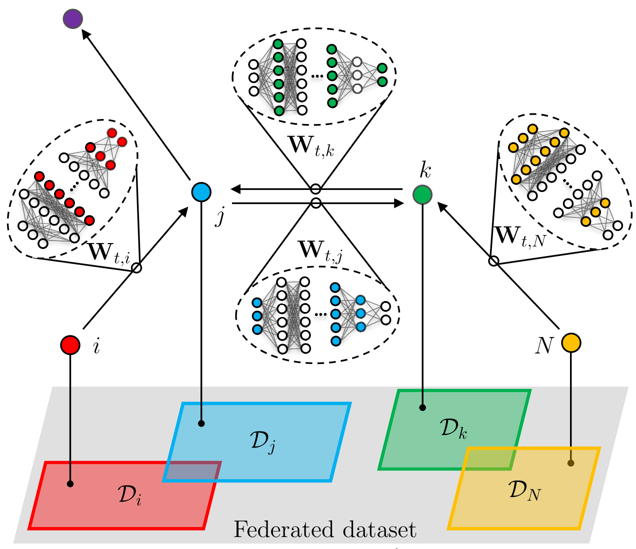


FIGURE 1. Decentralized FL system: cooperating agents autonomously share Machine Learning (ML) model fragments with neighbors and implement an average consensus strategy for model aggregation.

A promising methodology in this context is Federated Learning (FL) [18], [19], [20], [21], [22]. Rather than relying on raw data sharing as in standard Distributed Machine Learning (DML) tools [23], FL preserves the privacy as it requires only the exchange of locally trained parameters of a ML model, namely the weights and biases of a Neural Network (NN). Most of FL implementations use a Parameter Server (PS) to orchestrate the training process [19] while novel decentralized FL policies [24], [25], [26], [27], [28] exploit direct interactions among the agents. Average consensus typically serves as enabling technology to promote the decentralized fusion of the agent local models. The main issue is however the intensive utilization of sidelink network resources compared to vanilla FL methods. Indeed, as depicted in Fig. 1, each device is required to forward a copy of its own ML model to every neighbor, increasing the communication cost as the number of neighbors grows. Developing communication-efficient consensus tools is thus of fundamental importance, especially for bandwidth and energy-constrained devices.

A. RELATED WORKS AND MOTIVATIONS

First attempts to optimize the FL communication efficiency are mainly based on centralized architectures [29]. They select a suitable portion of the federated devices [30], [31], balance local and global model update frequencies [32], or employ quantization/sparsification operators [33], [34]. Only few approaches consider fully decentralized learning frameworks. For example, a layer-wise Alternating Direction Method of Multipliers (ADMM) is developed in [35] to save communication resources by communicating less frequently the largest layers. Other approaches propose to exchange a portion of the models [36], [37] or to exploit knowledge distillation [38]. Another proposed possibility [39] is to let devices perform more local optimization steps before each communication round.

In this paper, we propose a new approach for improving the communication efficiency specific for decentralized FL [24], [26]. The method is based on a *layer selection optimizer* that selects a number of relevant layer parameters to be shared among cooperating devices. Recent works have in fact demonstrated that applying compression in a layer-wise manner provides benefits compared to standard techniques operating directly on the full model [40], [41]. Indeed, different layers have different impact during the training process, and neglecting their importance when applying compression strategies may result in longer convergence times. This is especially important when considering large models possessing a large variability of trainable parameters across layers. It has been shown that layers exhibiting a large number of parameters generally encode the information in a redundant manner and therefore should be highly compressed or shared less frequently. In contrast, layers comprising few parameters typically show strong connections with preceding and succeeding layers. Thus, compression operators should be designed to specifically operate on these layers so as to not impact the final model performance. Layer-wise compression methods have been introduced relying on sparsification [40], [42] and randomized selection [41], [43]. However, the former requires repeating the sparsification operations on all layers, increasing the computational overhead as the number of layers grows, while the latter cannot capture any interrelation among layer parameters as it uses a simple randomized selection. In this paper, we propose to overcome these limits by a new combined approach that selects dynamically the most informative layers to be exchanged among the nodes based on both randomized and gradient-based selection strategies. The proposed method does not constrain the communication frequency of the largest layers as in [35], nor it preemptively divides the model into segments as in [36] and [37], but rather it selects only the most informative layers, according to the squared norm of the local gradients. Detailed contribution is summarized below.

B. PAPER CONTRIBUTIONS AND ORGANIZATION

We consider the decentralized FL system in Fig. 1 where a set of networked peer devices (or learners) cooperate to train a deep NN model. Each learner independently selects a subset of the NN layers and transmits the related trainable parameters (i.e., weights and biases) to the neighbors. Layer selection is implemented on each FL training round: the devices run an optimizer that sorts the layers of the NN model according to their expected contributions to the learning performance (e.g., measured by the squared norm of the gradients). The trainable parameters of the selected layers are then encoded for sidelink communication. The goal is to avoid the transmission of model parameters that may contribute minimally to the global model quality. The proposed strategy could also be extended to integrate quantization and pruning of the selected model parameters to further improve communication efficiency.

The proposed methods are first validated using the MNIST [44] dataset to assess latency, communication, and accuracy trade-offs with different connectivity patterns. Next, we consider the application of the developed FL policies to a cooperative sensing use case in vehicular scenarios. In the considered setup, vehicles rely on a complex NN, characterized by 40 trainable layers, to recognize road users/objects in their surroundings based on Lidar sensor readings. To extend the field of view of their ego-sensors, vehicles implement a FL optimization of the perception model via NN parameters sharing over Vehicle-to-Vehicle (V2V) links. Considering the vast amount of trainable layers and parameters of the ML model, the optimization of the information exchanged during the FL process is crucial so as to comply with limited sidelink resources. To summarize, the original contributions are as follows:

- A novel *fully-decentralized* FL system is proposed to target communication-constrained distributed ML implementations. The proposed architecture leverages average consensus and enables the agents to actively participate in the learning process by direct interactions via sidelinks.
- A parameter selection policy, referred to as Consensus-driven Federated Learning with Layer Selection (CFL-LS) is designed to select the most informative NN model parameters for transmission over the sidelinks. With this respect, we introduce a *layer optimizer* that selects a suitable population of the available model layers to be shared with neighbor nodes, based on local gradient observations.
- The impact of the CFL-LS policy on the consensus process is analyzed by considering different optimization and layer selection strategies.
- The approach is validated by extensive performance analysis in practical use cases, including connected automated driving.

Experimental results show that the proposed communication-efficient FL policy can reduce the communication resources up to 80% compared to standard FL setups that exchange all model parameters on every communication round. Effects of quantization, link loss/unavailability in wireless fading channels, and bandwidth constraints are also considered.

The paper is organized as follows. Sec. II describes the model of the proposed decentralized FL system. Sec. III presents the algorithms employed for layer selection, while Sec. IV analyzes the related convergence performance. The validation of the proposed method in image classification and vehicular sensing use cases is described in Sec. V and Sec. VI, respectively. Finally, Sec. VII draws the conclusions.

II. FEDERATED SYSTEM MODEL

The proposed FL setup consists of N interconnected agents, that mutually exchange the parameters of their local ML model optimized from local data samples via supervised learning methods. We assume that node i, with $i = 1, \ldots, N$, stores a dataset $\mathcal{D}_i = \{\mathbf{x}_h, \mathbf{y}_h\}_{h=1}^{E_i}$ composed by E_i training examples of the form $(\mathbf{x}_h, \mathbf{y}_h)$ where $\mathbf{x}_h$ is the input data while $\mathbf{y}_h$ is the corresponding desired prediction. The aggregated dataset is $\mathcal{D} = \bigcup_{i=1}^{N} \mathcal{D}_i$, with total number of examples $E = \sum_{i=1}^{N} E_i$. Local node datasets $\mathcal{D}_i$ are typically unbalanced, i.e., with varying sizes, and/or limited number of contained classes. A Deep Neural Network (DNN) is used to map the training data to the desired predictions. The DNN model is composed by L layers, with outputs $\mathbf{h}_\ell$ at layer ℓ computed by applying a non-linear (activation) function $f_\ell(.)$ to the weighted sum of the outputs of the previous layer $\mathbf{h}_{\ell-1}$ as

$$\mathbf{h}_\ell = f_\ell(\mathbf{w}_\ell^{\mathrm{T}} \mathbf{h}_{\ell-1} + \mathbf{b}_\ell), \tag{1}$$

where $\mathbf{w}_\ell$ and $\mathbf{b}_\ell$ are the weights and biases of layer ℓ, while for $\ell = 0$ we have $\mathbf{h}_0 = \mathbf{x}_h$ and $\mathbf{h}_L = \mathbf{y}_h$ for $\ell = L$. The weights and the biases of each layer can be conveniently aggregated into the matrix[1]

$$\mathbf{W} = [\mathbf{p}_1 \cdots \mathbf{p}_L]^{\mathrm{T}}, \tag{2}$$

where $\mathbf{p}_\ell = [\mathbf{w}_\ell^{\mathrm{T}}, \mathbf{b}_\ell^{\mathrm{T}}]^{\mathrm{T}} \in \mathbb{R}^{P_\ell \times 1}$ with $\ell = 1, \ldots, L$ is a compact vectorized representation of the weights and biases of layer ℓ with P_ℓ being the overall number of trainable parameters for the ℓ-th layer.

The goal of FL is to learn a global model $\mathbf{W}_\infty$, shared across all interconnected agents, for mapping the input data $\mathbf{x} = [\mathbf{x}_1 \cdots \mathbf{x}_E]^{\mathrm{T}}$ to the desired output predictions $\mathbf{y} = [\mathbf{y}_1 \cdots \mathbf{y}_E]^{\mathrm{T}}$ as best as possible. The global model parameters can be learned through the minimization of any finite-sum objective function $\mathcal{L}(\mathbf{W}_\infty)$ as

$$\mathbf{W}_\infty = \underset{\mathbf{W}}{\operatorname{argmin}}\, \mathcal{L}(\mathbf{W}) = \underset{\mathbf{W}}{\operatorname{argmin}} \underbrace{\sum_{i=1}^{N} \rho_i\, \mathcal{L}_i(\mathbf{W})}_{\mathcal{L}(\mathbf{W})}, \tag{3}$$

where $\rho_i = E_i/E$ and $\mathcal{L}_i(\mathbf{W}) = 1/E_i \sum_{h=1}^{E_i} \mathcal{L}_{i,h}(x_h, y_h; \mathbf{W})$ is the local loss of node i with $\mathcal{L}_{i,h}(\mathbf{x}_h, \mathbf{y}_h; \mathbf{W})$ being the loss computed over the example $(\mathbf{x}_h, \mathbf{y}_h)$ when $\mathbf{W}$ holds. The decentralized FL approach analyzed in the following relies on an average consensus policy to obtain $\mathbf{W}_\infty$ by repeatedly alternating the mutual exchange of local representations of the ML model

$$\mathbf{W}_{t,i} = [\mathbf{p}_{1,i}(t) \cdots \mathbf{p}_{L,i}(t)]^{\mathrm{T}} \tag{4}$$

on consecutive communication rounds $t = 0, 1, \ldots$, with local model optimization steps for minimizing the local loss $\mathcal{L}_i$.

A. COMMUNICATION MODEL

The model parameters $\mathbf{W}_{t,i}$ are exchanged by the agents to satisfy the half-duplex constraints: on each round, the devices multiplex a digital representation of the selected model parameters into a frame slot of T_F seconds and transmit

[1]This formulation considers that the parameters' dimensions do not change across layers. Nevertheless, the analysis can be easily extended/adapted to model variable number of parameters per layer. An example is given in Sec. VI.

such frame using orthogonal channels of bandwidth B_W. Connectivity among the agents is here represented as a undirected graph $\mathcal{G} = (\mathcal{V}, \mathcal{E})$, where $\mathcal{V}$ and $\mathcal{E}$ denote the set of nodes and edges, respectively.

At round t, the wireless link between a pair of devices $(k, i) \in \mathcal{E}$ at distance $d_{k,i}$ is assumed to be impaired by a frequency-flat time-varying fading channel with baseband complex-valued response $h_{k,i}(t) \sim \mathcal{CN}(0, 1)$, and instantaneous Signal-to-Noise Ratio (SNR)

$$\gamma_{k,i}(t) = \bar{\gamma}_0 S_0 \left(\frac{d_0}{d_{k,i}}\right)^{\upsilon} \left|h_{k,i}(t)\right|^2 \tag{5}$$

that accounts for the log-normal shadowing S_0, the path loss index υ, and the average SNR $\bar{\gamma}_0$ at reference distance d_0. A link is assigned as potential edge $(k, i) \in \mathcal{E}, \forall k \in \mathcal{N}_i$ of the graph $\mathcal{G}$ if $\gamma_{k,i}(t) > \beta$ where β is the receiver-side sensitivity threshold [45]. In what follows, we declare a link as unavailable with probability

$$\mathrm{P}_U = \mathrm{P_R}[\gamma_{k,i}(t) < \beta]. \tag{6}$$

The impact of link unavailability on FL under communication constraints is investigated in Sec. VI-C.

Device k sends the model updates encoded by $b_{k,t}$ bits according to the quantization scheme [46]. This encodes the model parameters in a stochastic manner by applying a randomized rounding operation that discretizes the parameters into a fixed set of levels (here varying between 256 and 1024, corresponding to $b_{k,t} = 8$ and $b_{k,t} = 10$ bits). Considering a frame/slot of T_F seconds, the number of bits chosen to encode the selected model parameters must satisfy the constraint

$$\frac{b_{k,t}}{T_F} < B_W \log[1 + \gamma_{k,i}(t)], \tag{7}$$

where $B_W \log[1 + \gamma_{k,i}(t)]$ is the link-layer spectral efficiency. Notice that the quantization process affects the time span (T_F) of each communication round (for an assigned efficiency) and thus the learning wall-clock time.

B. CONSENSUS-DRIVEN DECENTRALIZED FEDERATED LEARNING

On each FL round, the agent i fuses the local ML model $\mathbf{W}_{t,i}$ with the ones received from the set of available neighbors $\mathcal{N}_i$ as

$$\boldsymbol{\psi}_{t+1,i} = \mathbf{W}_{t,i} + \varepsilon \sum_{k \in \mathcal{N}_i} \boldsymbol{\Gamma}_k(t) \left(\mathbf{W}_{t,k} - \mathbf{W}_{t,i}\right), \tag{8}$$

where ε controls the stability of the consensus procedure [24]. The diagonal matrix

$$\boldsymbol{\Gamma}_k(t) = \sigma_{k,i} \operatorname{diag}[\mathbf{a}_k(t)], \tag{9}$$

with $\mathbf{a}_k(t) = [a_{1,k}(t) \cdots a_{L,k}(t)]^T$ contains the information about the layers that are chosen by the neighbors for transmission over the sidelinks. In particular, $\sigma_{k,i}$ is the mixing weight

$$\sigma_{k,i} = \frac{E_k}{\sum_{k \in \mathcal{N}_i} E_k}, \tag{10}$$

while $\mathbf{a}_k(t)$ is a binary vector encoding which layers have been transmitted by neighbor k. Each entry of $\mathbf{a}_k(t)$ is defined as:

$$a_{\ell,k}(t) = \begin{cases} 1 & \text{if } \mathbf{p}_{\ell,k}(t) \text{ is transmitted by node } k \\ 0 & \text{otherwise} \end{cases}. \tag{11}$$

Once the average consensus step is completed, the fused model is optimized locally using local data and a chosen optimizer.[2] Given a mini-batch $\mathcal{M}_i = \{\mathbf{x}_h, \mathbf{y}_h\}_{h=1}^B \subseteq \mathcal{D}_i$ of data, composed by B training examples, and assuming Adam optimization,[3] the aggregated model $\boldsymbol{\psi}_{t+1,i}$ in (8) can be updated through backpropagation as $\mathbf{W}_{t+1,i} = \boldsymbol{\psi}_{t+1,i} - \Delta\boldsymbol{\psi}_{t+1,i}$, with

$$\begin{cases} \Delta\boldsymbol{\psi}_{t+1,i} = \mu_t \cdot \dfrac{\sqrt{1-\beta_2^{\,t}}}{1-\beta_1^{\,t}} \cdot \dfrac{\mathbf{m}_{t+1,i}}{\sqrt{\mathbf{v}_{t+1,i}} + \delta} \\ \mathbf{m}_{t+1,i} = \beta_1 \mathbf{m}_{t,i} + (1-\beta_1)\nabla\mathcal{L}_i(\boldsymbol{\psi}_{t+1,i}|\mathcal{M}_i) \\ \mathbf{v}_{t+1,i} = \beta_2 \mathbf{v}_{t,i} + (1-\beta_2)\nabla^2\mathcal{L}_i(\boldsymbol{\psi}_{t+1,i}|\mathcal{M}_i), \end{cases} \tag{12}$$

where $\mathbf{m}_{t+1,i}$ and $\mathbf{v}_{t+1,i}$ are the first and second order moments of the gradients $\nabla\mathcal{L}_i(\boldsymbol{\psi}_{t+1,i}|\mathcal{M}_i) = [\nabla\mathbf{p}_{1,i}(t) \cdots \nabla\mathbf{p}_{L,i}(t)]^{\mathrm{T}}$ estimated with respect to the mean local loss $\mathcal{L}_i(\boldsymbol{\psi}_{t+1,i}|\mathcal{M}_i)$, averaged over the mini-batch $\mathcal{M}_i$. All parameters expressed in (12), i.e., $\beta_1, \beta_2 \in (0, 1]$, μ_t and δ are detailed in [47]. Finally, the updated model parameters $\mathbf{W}_{t+1,i}$ are forwarded to the neighbors of node i and a new communication round starts.

The decentralized FL procedure is iterated until local models reach a pre-defined target loss/accuracy or when they converge to the same representation, namely $\mathbf{W}_\infty$. The pseudo-code for the overall FL procedure is reported in Algorithm 1 which considers a more general gradient-based optimization.

III. LAYER SELECTION STRATEGIES

In this section, we propose a communication-efficient FL design that allows the agents joining the federation to select $M < L$ layers of the ML model to be shared with neighbors. The goal of the selection process is to provide a more efficient utilization of the communication bandwidth, such that the number of bits $b_{k,t} = b_{k,t}(M)$ chosen on each FL round satisfy (7) yet without penalizing convergence performance. Model accuracy and communication efficiency trade-offs are analyzed in two case studies described in Sec. V and Sec. VI.

The proposed method for the selection of the M layers of the NN is based on a *layer optimizer* that takes into account the local model and the data quality observed on each training round. In particular, in what follows we devise a policy for sorting the gradients of the local loss $\nabla\mathcal{L}_i(\boldsymbol{\psi}_{t+1,i}|\mathcal{M}_i)$ defined in (12), based on their squared running averages, as they provide an indicator about how informative each individual layer is, considering the local data/examples.

[2] Gradient-based optimizers are considered in the following analysis.

[3] Adam optimizer is adopted here as it typically requires little hyperparameter tuning [47] and is well suited for deep models.

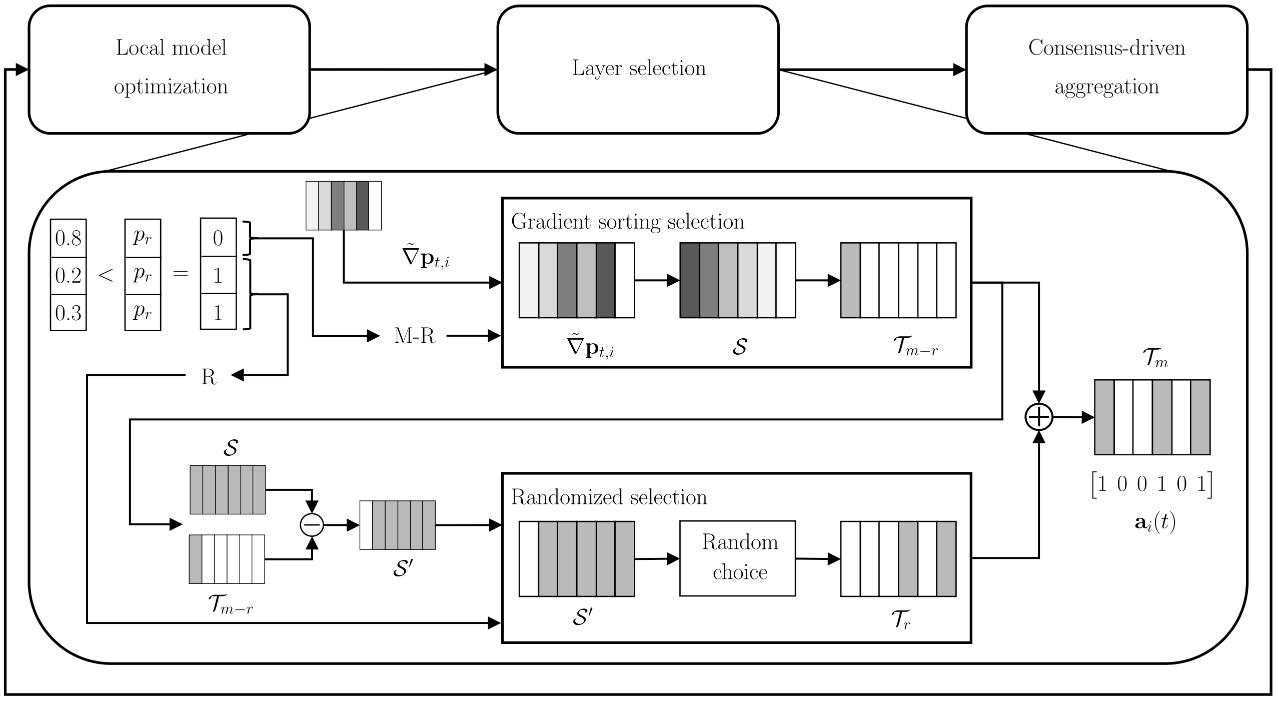


FIGURE 2. Layer selection process: each node runs a *layer optimizer* that sorts the layers according to their normalized squared gradient. Next, it combines $M-R$ layers selected from the optimizer with other R chosen randomly. The overall selected layers are then forwarded to the neighbors for average consensus.

Selected layers and corresponding model parameters are then exchanged in the consensus step.

Given a measure of the local gradient $\nabla\mathcal{L}_i(\boldsymbol{\psi}_{t+1,i}|\mathcal{M}_i)$, estimated by node i using a mini-batch $\mathcal{M}_i$ of its local data, we first compute the average over the available mini-batches

$$\bar{\nabla}\mathcal{L}_i(\boldsymbol{\psi}_{t+1,i}) = \frac{1}{E_i}\sum_{\mathcal{M}_i\in\mathcal{D}_i}\nabla\mathcal{L}_i(\boldsymbol{\psi}_{t+1,i}|\mathcal{M}_i), \tag{13}$$

with $\bar{\nabla}\mathcal{L}_i(\boldsymbol{\psi}_{t+1,i}) = [\bar{\nabla}\mathbf{p}_{1,i}(t) \cdots \bar{\nabla}\mathbf{p}_{L,i}(t)]^{\mathrm{T}}$ where $\bar{\nabla}\mathbf{p}_{\ell,i}(t) = 1/E_i\sum_{\mathcal{M}_i\in\mathcal{D}_i}\nabla\mathbf{p}_{\ell,i}(t)$ collects the vectorized gradients of layer ℓ, averaged over the available mini-batches. Next, we compute the squared norm of the gradients with respect to the trainable parameters characterizing each NN layer

$$g_{\ell,i}(t) = \frac{1}{P_\ell}\|\bar{\nabla}\mathbf{p}_{\ell,i}(t)\|^2. \tag{14}$$

Let $\mathcal{S} = (\ell_1,\ldots,\ell_L)$ be an ordered set that contains the layer indices ℓ_i sorted in descending order according to the gradient measures contained in

$$\mathbf{g}_{t,i} = [g_{1,i}(t)\cdots g_{L,i}(t)]^{\mathrm{T}}, \tag{15}$$

we construct a subset $\mathcal{T}_m = (\ell_k : 1 \le k \le M) \subseteq \mathcal{S}$ of the M chosen layers with the largest gradient metric $g_{\ell,i}(t)$. The elements of $\mathbf{a}_i(t)$ are thus defined as

$$a_{i,\ell}(t) = \begin{cases} 1 & \text{if } \ell\in\mathcal{T}_m \\ 0 & \text{otherwise} \end{cases} \tag{16}$$

and subject to $\mathbf{1}^T\mathbf{a}_i(t) < M$. Once $\mathbf{a}_i(t)$ has been constructed, it is transmitted along with the selected layers during the consensus step. Nodes receiving the model updates are then able to retrieve $\boldsymbol{\Gamma}_i(t)$ from $\mathbf{a}_i(t)$ and fuse the received parameters according to (8). Note that the layer selection process is performed locally at each device and does not require any information from neighbors. In particular, we keep track of the gradients estimated during the local optimization step and sort them in descending order. Intuitively, layers exhibiting higher gradients convey more information about the local data and should be therefore selected and propagated to neighbors. A comparative analysis with other solutions is given in Sec. V-B.

Note that gradient sorting methodology for layer selection might cause the parameters of some layers to be never, or rarely, exchanged.[4] As analyzed in the following, this may negatively affect the overall convergence of the FL process. To overcome this limitation, we propose to alternate gradient sorting with a randomized layer selection policy: the goal is to let the nodes receive a fair share of the neighbor NN model layers over consecutive rounds. In particular, we consider R out of M layers as being chosen randomly on each FL round,

$$R = \sum_{n=1}^{M} 1_{u_n<p_r} \tag{17}$$

with $1_{u_n<p_r}$ the unit step function, $u_n \sim \mathcal{U}(0,1)$ and p_r is the probability of selecting a layer randomly. We select $M-R$ elements from the ordered set $\mathcal{S}$ to construct $\mathcal{T}_{m-r} = (\ell_k : 1 \le k \le (M-R)) \subset \mathcal{S}$ while the last R elements are drawn

[4]for example, this corresponds to a case where $\mathbf{g}_{t,i}$ metric remains static for many consecutive training rounds.

Algorithm 1 Consensus-Driven Federated Learning With Layer Selection (CFL-LS)

```
1:  procedure CFL-LS($\mathcal{N}_i, \sigma, M, p_r$)
2:      initialize $\mathbf{W}_{0,i} \leftarrow$ node $i$
3:      initialize $\mathbf{m}_{0,i} \leftarrow 0$
4:      initialize $\mathbf{v}_{0,i} \leftarrow 0$
5:      for each round $t = 1, 2, \ldots$ do                    ▷ Training loop
6:          receive$\{\mathbf{W}_{t,k}, \mathbf{a}_k(t)\}_{k\in\mathcal{N}_i}$     ▷ RX model
7:          $\boldsymbol{\psi}_{t+1,i} \leftarrow \mathbf{W}_{t,i}$
8:          for all nodes $k \in \mathcal{N}_i$ do
9:              $\boldsymbol{\Gamma}_k(t) \leftarrow \sigma_{k,i}\mathrm{diag}[\mathbf{a}_k(t)]$
10:             $\boldsymbol{\psi}_{t+1,i} \leftarrow \boldsymbol{\psi}_{t+1,i} + \varepsilon\boldsymbol{\Gamma}_k(t)\ (\mathbf{W}_{t,k} - \mathbf{W}_{t,i})$
                                                                ▷ Consensus
11:         end for
12:         $\mathbf{W}_{t+1,i}, \bar{\nabla}\mathcal{L}_i(\boldsymbol{\psi}_{t+1,i}) =$ ModelUpdate($\boldsymbol{\psi}_{t+1,i}$)
13:         $\mathbf{a}_i(t) =$ LayerSelection($\bar{\nabla}\mathcal{L}_i(\boldsymbol{\psi}_{t+1,i}), M, p_r$)
14:         send$\big(\mathbf{W}_{t+1,i}, \mathbf{a}_i(t)\big)$              ▷ TX to neighbors
15:     end for
16: end procedure
17: procedure ModelUpdate($\boldsymbol{\psi}_{t+1,i}$)
18:     initialize $\bar{\nabla}\mathcal{L}_i(\boldsymbol{\psi}_{t+1,i}) \leftarrow 0$
19:     $\mathcal{B}_i \leftarrow$ mini-batches of size B
20:     for batch $\mathcal{M}_i \in \mathcal{B}_i$ do
21:         $\boldsymbol{\psi}_{t+1,i}, \nabla\mathcal{L}_i(\boldsymbol{\psi}_{t+1,i}|\mathcal{M}_i) =$
              GradOptimizer($\boldsymbol{\psi}_{t+1,i}$)
22:         $\bar{\nabla}\mathcal{L}_i(\boldsymbol{\psi}_{t+1,i}) \leftarrow \bar{\nabla}\mathcal{L}_i(\boldsymbol{\psi}_{t+1,i})+$
              $\nabla\mathcal{L}_i(\boldsymbol{\psi}_{t+1,i}|\mathcal{M}_i)$
23:     end for
24:     $\mathbf{W}_{t+1,i} \leftarrow \boldsymbol{\psi}_{t+1,i}$
25:     $\bar{\nabla}\mathcal{L}_i(\boldsymbol{\psi}_{t+1,i}) \leftarrow \frac{1}{E_i}\bar{\nabla}\mathcal{L}_i(\boldsymbol{\psi}_{t+1,i})$ ▷ Average gradients
26: end procedure
```

Algorithm 2 Layer Selection Policy

```
1:  procedure LayerSelection($\bar{\nabla}\mathcal{L}_i(\boldsymbol{\psi}_{t+1,i}), M, p_r$)
2:      for each $n = 1, \ldots, M$ do
3:          $u_n \sim \mathcal{U}(0,1)$
4:      end for
5:      $R \leftarrow \sum_{n=1}^{M} 1_{u_n<p_r}$
6:      for each layer $\ell = 1, \ldots, L$ do
7:          $g_{\ell,i}(t) \leftarrow \frac{1}{P_\ell}\|\bar{\nabla}\mathbf{p}_{\ell,i}(t)\|^2$
8:      end for
9:      $\mathbf{g}_{t,i} \leftarrow [g_{1,i}(t), \ldots, g_{L,i}(t)]^{\mathrm{T}}$
10:     $\mathcal{S} \leftarrow$ sort($\mathbf{g}_{t,i}$)
11:     $\mathcal{T}_{m-r} \leftarrow (\ell_k : 1 \le k \le (M-R)) \subset \mathcal{S}$
12:     $\mathcal{S}' \leftarrow \mathcal{S} \setminus \mathcal{T}_{m-r}$
13:     $\mathcal{T}_r \leftarrow \{\ell_k : 1 \le k \le R\} \subset \mathcal{S}'$
14:     $\mathcal{T}_m \leftarrow \mathcal{T}_{m-r} \bigcup \mathcal{T}_r$
15:     for $\ell = 1, \ldots, L$ do
16:         $a_{i,\ell}(t) \leftarrow \begin{cases} 1 & \text{if } \ell \in \mathcal{T}_m \text{ and } \mathbf{1}^{\mathrm{T}}\mathbf{a}_i(t) < M \\ 0 & \text{otherwise} \end{cases}$
17:     end for
18:     $\mathbf{a}_i(t) \leftarrow [a_{i,1}(t), \ldots, a_{i,L}(t)]^{\mathrm{T}}$
19: end procedure
```

randomly from $\mathcal{S}' = \mathcal{S} \setminus \mathcal{T}_{m-r}$ to obtain $\mathcal{T}_r = \{\ell_k : 1 \le k \le R\} \subset \mathcal{S}'$. The layers and the corresponding model parameters selected for sidelink transmission thus belong to the set $\mathcal{T}_m = \mathcal{T}_{m-r} \bigcup \mathcal{T}_r$. The full layer selection policy, depicted in Fig. 2, integrating the gradient sorting and the randomized approach, is reported in Algorithm 2.

IV. IMPACT OF LAYER SELECTION ON CONSENSUS

This section analyzes the impact of the proposed layer selection methods on the FL convergence. The goal is to study the minimal (and necessary) conditions for which the consensus process converges to the average of the models.

First, we assume that the local loss functions $\mathcal{L}_i(\mathbf{W})$ are smooth with constant $L > 0$, namely their gradients $\nabla\mathcal{L}_i(\mathbf{W})$ are Lipschitz continuous with constant L, and μ-strongly convex [20], while the Adam optimizer step-size μ_t is properly chosen so that the objective functions $\mathcal{L}_i(\mathbf{W})$ are decreasing with each Adam iteration (i.e., after some threshold). Considering the average consensus aggregation model of (8), we derive the conditions for convergence under the assumption that each client adopts the layer selection optimizer analyzed in Sec. III. To simplify the reasoning we also assume that the number of model parameters does not vary across layers, i.e., $P_\ell = P$, the mixing weights are the same for each client $\sigma_k = \sigma$, namely each client has the same number of examples E and uses the same number of neighbors for model aggregation.[5] We rewrite (8) into

$$\boldsymbol{\psi}_{t+1,i} = \left(\mathbf{I}_M - \varepsilon\sum_{k\in\mathcal{N}_i}\boldsymbol{\Gamma}_k(t)\right)\mathbf{W}_{t,i} + \varepsilon\sum_{k\in\mathcal{N}_i}\boldsymbol{\Gamma}_k(t)\cdot\mathbf{W}_{t,k}, \tag{18}$$

now with $\boldsymbol{\Gamma}_k(t) = \sigma\cdot\mathrm{diag}\left[\mathbf{a}_k(t)\right]$ as in (9). Next, considering the Adam optimization and the consensus process we obtain

$$\mathbf{W}_{t+1,i} = \underbrace{\sum_{k\in\mathcal{N}_i\cup i}\widetilde{\boldsymbol{\Gamma}}_{i,k}(t)\cdot\mathbf{W}_{t,k}}_{\psi_{t+1,i}} - \Delta\boldsymbol{\psi}_{t+1,i}, \tag{19}$$

with Adam update $\Delta\boldsymbol{\psi}_{t+1,i}$ in (12), $\widetilde{\boldsymbol{\Gamma}}_{i,k}(t) = \varepsilon\boldsymbol{\Gamma}_k(t)$, for $k \in \mathcal{N}_i$, and $\widetilde{\boldsymbol{\Gamma}}_{k,k}(t) = \mathbf{I}_M - \varepsilon\sum_{k\in\mathcal{N}_i}\boldsymbol{\Gamma}_k(t)$. We collect all the N local estimates (from all the N clients) of the L model layers into the $LN \times P$ matrix $\mathcal{W}_t = \left[\mathbf{W}_{t,1}^{\mathrm{T}} \cdots \mathbf{W}_{t,N}^{\mathrm{T}}\right]^{\mathrm{T}}$. Consensus-driven model aggregation of (19) can be thus further rewritten as

$$\mathcal{W}_{t+1} = \underbrace{(\mathbf{I}_{KM} - \varepsilon\mathbf{L}_{FL})}_{\mathbf{P}}\mathcal{W}_t - \Delta\Psi(\mathbf{P}\mathcal{W}_t) \tag{20}$$

[5]Notice that the number of neighbors is typically pre-determined during FL initialization and corresponds to a fixed-size subset of the neighbors within the communication range.

with the $LN \times P$ matrix $\Delta\Psi(\mathbf{P}\mathcal{W}_t)$ that contains the N Adam updates of size $L \times P$

$$\Delta\Psi(\mathbf{P}\mathcal{W}_t) = \left[\Delta\boldsymbol{\psi}_{t+1,1}^{\mathrm{T}} \cdots \Delta\boldsymbol{\psi}_{t+1,N}^{\mathrm{T}}\right]^{\mathrm{T}}. \quad (21)$$

$\mathbf{P}$ and $\mathbf{L}_{FL}$ in (20) are the Perron and the Laplacian matrices. In particular, the Laplacian $\mathbf{L}_{FL} = [\mathbf{L}_{FL}(i,k),\ i,k = 1,\ldots,N]$ has dimension $LN \times LN$ and it is partitioned into $N \times N$ blocks $\mathbf{L}_{FL}(i,k)$ of dimension $L \times L$. Each block is defined as

$$\mathbf{L}_{FL}(i,k) = \begin{cases} -\boldsymbol{\Gamma}_k(t) & k \in \mathcal{N}_i \\ \sum_{k\in\mathcal{N}_i} \boldsymbol{\Gamma}_k(t) & k = i \\ \mathbf{0} & \text{otherwise.} \end{cases} \quad (22)$$

It can be observed from (9) and (22) that the layer selections $\mathbf{a}_i(t)$ made by individual clients $i = 1,\ldots,N$ modify the weight matrix $\boldsymbol{\Gamma}_k(t)$ and thus the Laplacian $\mathbf{L}_{FL}$. Let $\bar{\mathbf{a}}(t) = \left[\mathbf{a}_1^T(t)\cdots\mathbf{a}_N^T(t)\right]^T$ represent the layer selections at time t, made independently by each client based on gradient sorting, the consensus equation at time t becomes

$$\mathcal{W}_{t+1} = \mathbf{P}_{\bar{\mathbf{a}}(t)}\mathcal{W}_t, \quad (23)$$

with $\mathbf{P}_{\bar{\mathbf{a}}(t)} = \mathbf{I}_{KM} - \varepsilon\mathbf{L}_{FL}[\bar{\mathbf{a}}(t)] \in \mathcal{P}$ and $\mathcal{P}$ collecting the finite set of all the possible layer choices. The consensus process at discrete times $q = t_0,\ldots,t$ can be thus written from (20) as

$$\mathcal{W}_t = \left(\prod_{q=t_0}^{t}\mathbf{P}_{\bar{\mathbf{a}}(q)}\right)\mathcal{W}_{t_0} = \Lambda_t \cdot \mathcal{W}_{t_0}, \quad (24)$$

with $\mathcal{W}_{t_0} = \left[\mathbf{W}_{t_0,1}^{\mathrm{T}}\cdots\mathbf{W}_{t_0,N}^{\mathrm{T}}\right]^{\mathrm{T}}$ being a set of local models obtained at time t_0 by the Adam optimizer and $\Lambda_t = \prod_{q=t_0}^{t}\mathbf{P}_{\bar{\mathbf{a}}(q)}$. The vector $\bar{\mathbf{a}}(q)$ can be regarded as a switching signal of the discrete-time system (24) since it can assume any value in the finite set $\mathcal{P}$. According to the convergence properties of switched consensus systems [48], [49], the average consensus process converges when the (infinite) product of the stochastic matrices $\{\mathbf{P}_{\bar{\mathbf{a}}(q)}\}, q = t_0,\ldots,t$ for $t \to +\infty$ has a limit. In the following we exploit the above result to assess the convergence of the layer selection policies.

A. LAYER SELECTION POLICIES AND CONVERGENCE

Recalling that $\bar{\mathbf{a}}(t)$ is a switching signal in the (finite) set $\mathcal{P}$, we analyze the convergence of two selected policies from Sect. III.

1) POLICY #1: SELECTION W/O COORDINATION

This strategy lets the chosen layers to be selected independently by every device in each round, without any coordination. Under this policy with $p_r \in (0,1]$, the matrix product $\prod_{q=t_0}^{t}\mathbf{P}_{\bar{\mathbf{a}}(q)}$ in (24) is ergodic, however the sequence $\mathbf{P}_{\bar{\mathbf{a}}(t)},\ldots,\mathbf{P}_{\bar{\mathbf{a}}(t_0)}$ is not composed by doubly stochastic matrices. As a result, the consensus process does not converge to the weighted average of the initial local models [50].

2) POLICY #2: SELECTION WITH COORDINATION

In this second scenario we constrain the devices to agree on a sequence of layers to be exchanged for each communication round. The sequence is random when $p_r = 1$. Coordination among devices can be done at the start of the training process via a coordinator device that selects the overall sequence of selected layers that will be exchanged by all devices members of the federation for the assigned communication rounds. By adopting this strategy, the Laplacian matrix $\mathbf{L}_{FL}$ in (22) becomes symmetric and satisfies $\mathbf{L}_{FL}\mathbf{1}_{LN} = \mathbf{0}$, $\mathbf{1}_{LN}^{\mathrm{T}}\mathbf{L}_{FL} = \mathbf{0}$, making $\mathbf{P}_{\bar{\mathbf{a}}(q)}$ doubly stochastic. This holds for all rounds, i.e., for $q = t_0,\ldots,t$ with $t \to +\infty$. As a result, there exists a finite vector $\boldsymbol{\alpha}$ for which the limit

$$\lim_{t\to\infty}\prod_{q=t_0}^{t}\mathbf{P}_{\bar{\mathbf{a}}(q)} = \mathbf{1}\cdot\boldsymbol{\alpha}^T \quad (25)$$

holds [48], [49]. As proved in the Appendix A, the consensus process converges to the average values of the initial local models $\mathcal{W}_{t_0}$ [50].

Remark: Setting $p_r = 0$ produces layers that are approximately time-invariant over consecutive FL rounds, therefore $\bar{\mathbf{a}}(t) = \bar{\mathbf{a}}$ as experimentally verified in Sec. V.

In this case (24) becomes a linear, time-invariant, discrete-time system. Therefore, the consensus process converges to

$$\lim_{t\to\infty}\mathbf{W}_{t,i} = \left(\sum_{m=1}^{N}\boldsymbol{\Gamma}_m\right)^{-1}\sum_k \boldsymbol{\Gamma}_k\mathbf{W}_{0,k} \quad \forall i \in \mathcal{V}, \quad (26)$$

as shown in [4]. This holds for all aforementioned policies.

B. CONVERGENCE RESULTS

To experimentally verify the convergence of the consensus process derived in Sec. IV-A, here we analyze Policy #1 and Policy #2, both with $p_r = 1$ (i.e., for random selection performed on all layers). We employ a NN composed by $L = 6$ layers each containing a single trainable parameter. $N = 10$ devices participate to the consensus process with an all-to-all connectivity. During the consensus stage, each device exchanges $M = 2$ layers out of the total L. Results are reported focusing on the effect of layer sharing policies on consensus.

Fig. 3a reports the convergence results for Policy #1 while Fig. 3b refers to Policy #2. Both figures show how the NN parameters evolve during the consensus steps for all devices, while the average of the local models is depicted in dashed black line. The analysis confirms that Policy #2 convergences to the average of the initial models while Policy #1 does not as each NN parameter $\mathbf{p}_k$ with $k = 1,\ldots,6$ converges to a different value that depends on the sequence of layers selected by the devices during the FL process. Furthermore, the coordinator-based selection strategy is seen to require a larger number of communication rounds to converge. On the other hand, Policy #1 converges quite quickly while the other strategy requires more consensus steps, indicating that constraining the selected layers to be the same among all devices, as done by Policy #2, may result in longer training times. To conclude the analysis, in Fig. 3c we evaluate how

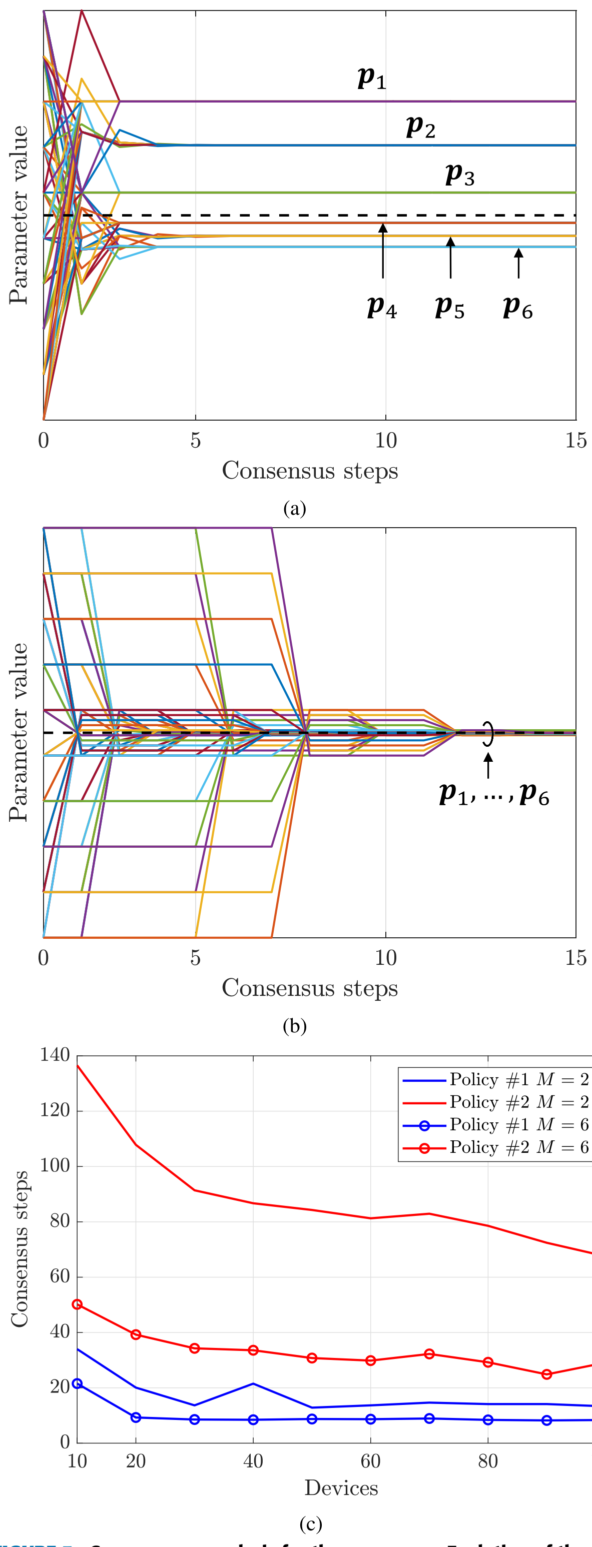


FIGURE 3. Convergence analysis for the consensus. Evolution of the NN parameters versus the iteration for Policy #1 (a) and Policy #2 (b), while (c) shows the aggregate results for different selected layers and number of devices.

many consensus steps are needed for reaching convergence by varying the number of devices between 10 and 100, and considering $M = \{2, 6\}$. The ML model now is constituted by $L = 20$ trainable layers each containing a single parameter. These last results further confirm the superior convergence properties of Policy #1, especially when few devices participate to the consensus process and for low values of M.

TABLE 1. MNIST architecture.

Layer (number)	Filters/Neurons	Parameters	Size (Kb)
CL1 (1)	16	160	5.12
CL2 (2)	32	4640	148.48
CL3 (3)	32	9248	295.94
FC1 (4)	32	1056	33.79
FC2 (5)	32	1056	33.69
FC3 (6)	10	330	10.56

V. VALIDATION WITH MNIST DATA

In this section, we validate the proposed CFL-LS approach (Sec. III) considering a classification task with the benchmark MNIST dataset [44]. Several baseline methods are used as comparison to show the benefits of the developed layer selection strategies. Sec. V-A details the main simulation parameters employed for assessing the performances of the developed techniques, while Sec. V-B shows a first validation of the proposed method, by comparing different gradient sorting approaches and studying how the training process is affected if (some) layers are never or rarely transmitted. Sec. V-C provides a more in-depth investigation of latency, accuracy, and communication cost trade-offs. More specifically, we analyze the performances of the proposed approach for varying number of transmitted layers, comparing them also against a centralized FL solution and a DML implementation. Then, we evaluate the differences between a decentralized and centralized FL tool for the case where both methods employ the layer selection strategies presented in Sec. III. Finally, Sec. V-D studies the effects of quantization procedures applied to the selected layers and how they affect the final performances.

A. SYSTEM PARAMETERS

The overall FL process is deployed into a virtual platform that allows to configure the devices as distributed local learners and to support device-to-device (D2D) interactions. In particular, in this initial example we consider a ring network of $N = 10$ agents each connected to a varying and configurable number of contiguous neighbors. We analyze the performance of three connectivity patterns, corresponding to agents connected to the 10%, 50% and 90% of all the possible devices, respectively. These patterns model sparse (10%) to dense (90%) networking scenarios.

For the considered FL setup, each agent is assigned 300 randomly drawn MNIST training examples for 6 classes out of 10, to simulate non-independent and identically distributed (non-iid) information across the devices. The ML model employed by each device and the size of the trainable parameters for each layer are reported in Table 1. In total, the number of parameters of the NN is 16490 and the number of trainable layers is $L = 6$. Mini-batch Adam optimization is used for updating the local model according to (12) with $B = 30$ examples, and with parameters $\mu_t = 5 \cdot 10^{-4}$, $\beta_1 = 0.9$, $\beta_2 = 0.999$ and $\delta = 10^{-7}$. At the end of each communication round, the performances for all agents are computed using the full MNIST validation dataset.

The proposed CFL-LS method is first evaluated by varying the communication bandwidth constraint, or equivalently the link-layer spectral efficiency of (7). More specifically, assuming a frame duration $T_F \approx 45$ ms and $b_{k,t} = 32$ bits, we analyze the performances with B_W ranging from 2 to 10 MHz. This corresponds to the exchange of $M = 1$ up to $M = 4$ layers. In these examples we also assume $P_U = 0$, however, the effect of link unavailability $P_U > 0$ is considered in the following cases. The performance of CFL-LS is assessed and compared against several baseline approaches: i) the classical FL solution, where a PS collects the updated models from all the available devices and layers of the NN; ii) a DML implementation that fuses the raw data at a data center, and iii) a conventional decentralized FL policy, referred to as Consensus-driven Federated Learning (CFL), where all the model parameters are shared (i.e., $M = 6$) by all the interconnected agents over an all-to-all connectivity network. Note that the aforementioned methods employed for comparison are not subject to bandwidth constraints.

B. ASCENDING VS. DESCENDING GRADIENT SORTING COMPARISON

In Fig. 4 we compare two different strategies for gradient sorting and layer selection. The first one sorts the layers of the NN in a descending order (descending gradient sorting, DGS) while the second one uses an ascending order (AGS). Descending ordering prevents the transmission of layers with low gradients $\mathbf{g}_{t,i}$, while ascending ordering favors the transmission of these layers. These strategies are denoted as AGS $p_r = 0$ and DGS $p_r = 0$, as relying only on gradient sorting operations. The performances are also studied for the case where AGS and DGS integrate a randomized layer selection by using the strategy presented in Sec. III with probability threshold $p_r = 0.2$, here referred to as AGS $p_r = 0.2$ and DGS $p_r = 0.2$. All strategies are compared with devices sharing $M = 4$ layers. As concerns the D2D connection, we evaluate the layer selection policies over the 50% connectivity scenario. Validation loss and accuracy are used as performance metrics, averaged over all participating devices and over all runs.

Fig. 4a reports the validation loss for the aforementioned layer selection strategies, while Fig. 4b shows the percentage of times each layer is transmitted during the FL process. Comparing the results, DGS shows far superior performances for all cases when compared with AGS. With $p_r = 0$, AGS exhibits extremely low convergence properties while DGS reaches the minimum of its validation loss curve within 100 communication rounds. Nevertheless, even though DGS is much more rapid to converge, it shows clear signs of overfitting after 100 communication rounds. This may be related to the number of times each layer is exchanged by

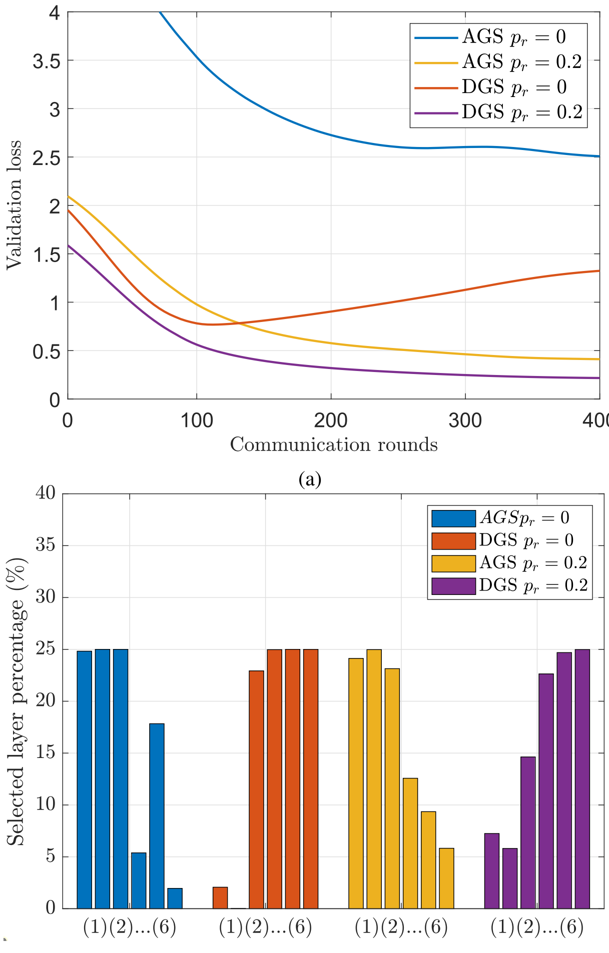


FIGURE 4. Analysis of the DGS and ADS strategies for layer selection: (a) validation loss, (b) selected layer percentage. Layers(x) for x = 1, . . . , 6 are defined in Table 1.

the devices, as reported in Fig. 4b, where the 4th and 6th layers are selected quite rarely. By allowing a more fair layer exchange, i.e., when $p_r = 0.2$, performances can be heavily improved both for AGS and DGS while also avoiding overfitting problems. This indicates that not transmitting layers for long time periods or excluding some of them entirely from being sent during training heavily impacts the learning performances.

C. LAYER SELECTION POLICY ASSESSMENT

In this section, we show that a partial random selection of the layers, regardless of gradient sorting, allows to train higher-quality models. The validation focuses on three different threshold probability p_r values namely $p_r = 0.2$, $p_r = 0.6$ and $p_r = 1.0$, and considers the connectivity patterns defined in Sec. V-A.

Fig. 5 reports the validation loss (top) and validation accuracy (bottom) obtained by sharing a number of layers of the NN per round equal to $M = 1$ (Fig. 5a and 5d), $M = 2$ (Fig. 5b and 5e) and $M = 4$ (Fig. 5c and 5f). The comparison considers also the centralized (FL), the consensus-driven (CFL) and the DML scheme. Focusing on

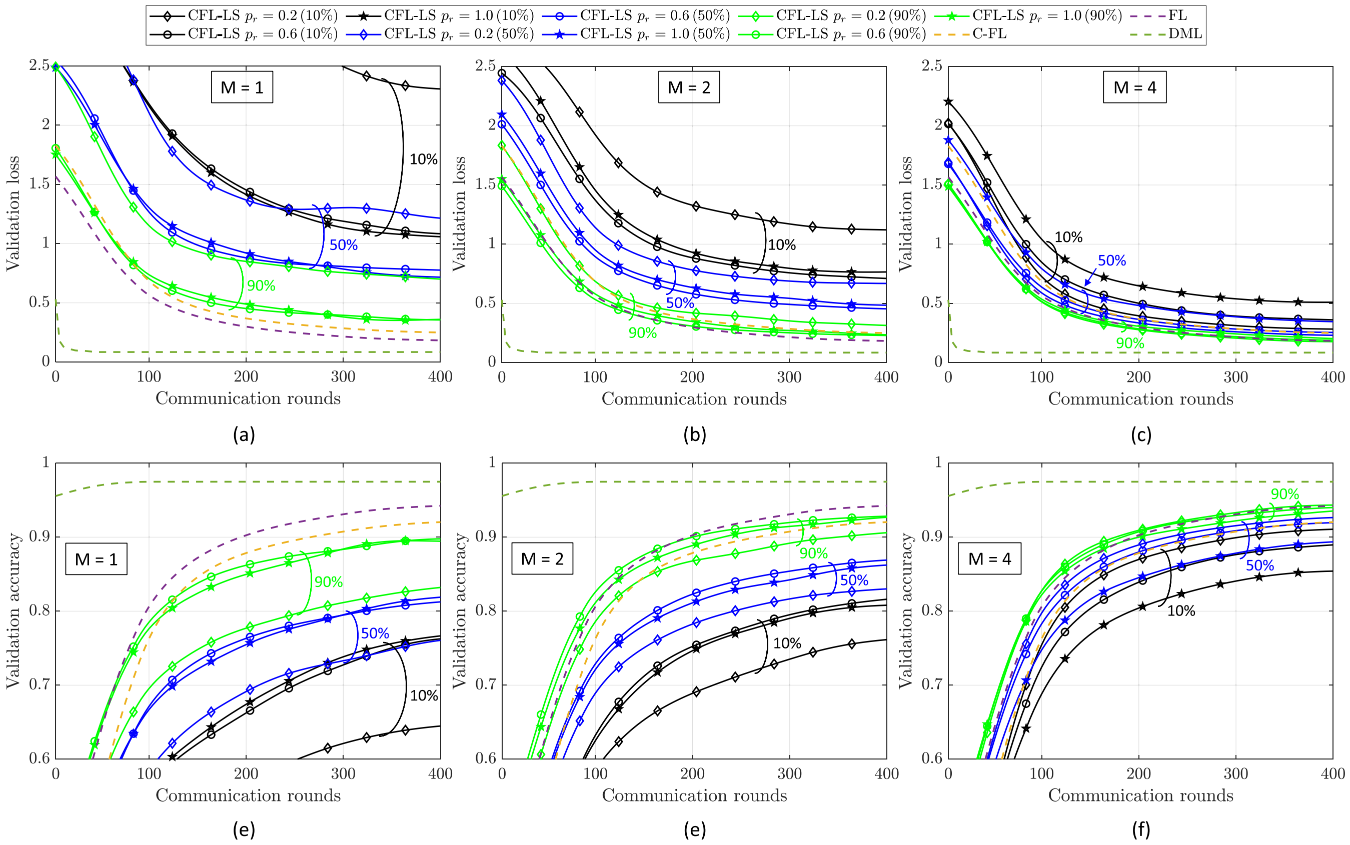


FIGURE 5. Performance analysis of the CFL-LS method in terms of validation loss (top) and accuracy (bottom) for number of shared layers equal to (a,d) $M = 1$, (b,e) $M = 2$ and (c,f) $M = 4$. Percentages indicate which connectivity pattern is considered among 10%, 50% and 90% cases. Validation loss/accuracy metrics are averaged over 10 independent runs and over all devices.

TABLE 2. Comparison of the validation accuracy and validation loss obtained with the MNIST dataset for all methods considered.

Layers	DML	FL	C-FL	CFL-LS								
				p_r (10 % connectivity)			p_r (50 % connectivity)			p_r (90 % connectivity)		
				0.2	0.6	1.0	0.2	0.6	1.0	0.2	0.6	1.0
$M = 1$ (val. loss)	**0.084**	**0.182**	**0.249**	2.306	1.083	**1.058**	1.214	0.776	**0.716**	0.700	0.360	**0.355**
$M = 2$ (val. loss)	**0.084**	**0.182**	**0.249**	1.120	**0.710**	0.765	0.668	**0.454**	0.484	0.313	**0.228**	0.234
$M = 4$ (val. loss)	**0.084**	**0.182**	**0.249**	**0.282**	0.360	0.509	**0.230**	0.254	0.345	**0.176**	0.188	0.202
$M = 1$ (val accuracy)	**0.974**	**0.942**	**0.920**	0.644	0.762	**0.766**	0.760	0.812	**0.818**	0.832	0.894	**0.897**
$M = 2$ (val. accuracy)	**0.974**	**0.942**	**0.920**	0.761	**0.816**	0.808	0.830	**0.869**	0.862	0.906	**0.928**	0.926
$M = 4$ (val. accuracy)	**0.974**	**0.942**	**0.920**	**0.910**	0.889	0.854	**0.926**	0.919	0.893	**0.943**	0.940	0.935

the performances for varying number M of shared layers, it can be noticed that carefully selecting p_r can be beneficial in reducing the communication resources without introducing accuracy penalties. Indeed, choosing $p_r = 1.0$ when $M = 1$ instead of $p_r = 0.2$ when $M = 2$ allows to halve the number of layers exchanged without altering the final performances. Similarly, selecting $p_r = 0.6$ when $M = 2$ rather than $p_r = 1.0$ when $M = 4$ decreases the communication cost. Overall, the proposed approach can reach almost the same performances of CFL and FL for all M values when the probability threshold is adequately tuned. Interestingly, in some cases sending all model parameters on each round, as done in conventional CFL and FL policies, may not be the optimal choice for reducing quickly the validation loss. For example, this is verified in dense connectivity patterns (90% connectivity) as the validation loss obtained at the start of the training process is lower compared to CFL and FL tools.

Considering now the performances for varying threshold probabilities p_r, the results indicate that balancing gradient sorting and randomized selection operations is fundamental and should be performed according to the available communication resources, quantified here by M. With $M = 1$, high values of p_r should be preferred since they allow to obtain far superior performances compared to selection policies relying entirely on gradient sorting functions. This is confirmed by the validation loss/accuracy gap between $p_r = 0.2$, $p_r = 0.6$ and $p_r = 1.0$ for all connectivity patterns. On the other

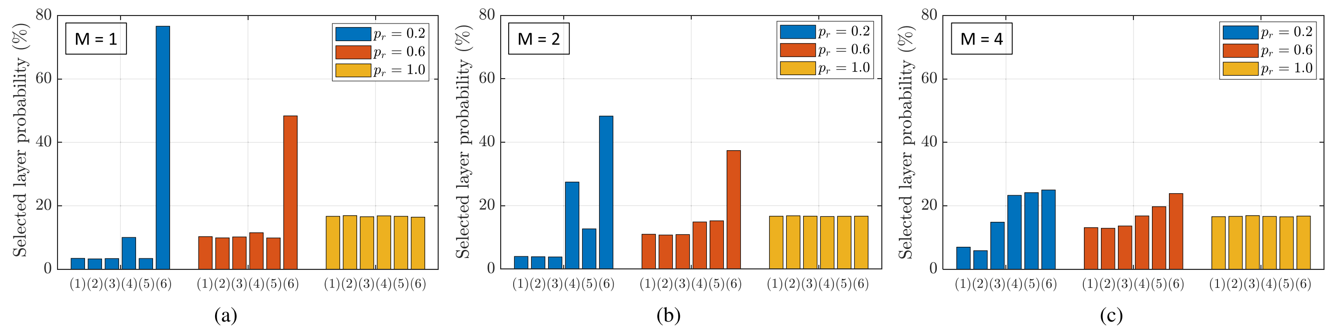


FIGURE 6. Probability of selected ML model layers for MNIST processing: analysis of the layer exchanged among the cooperating agents with (a) $M = 1$, (b) $M = 2$ and (c) $M = 4$. Layers(x) for $x = 1, \ldots, 6$ are defined in Table 1.

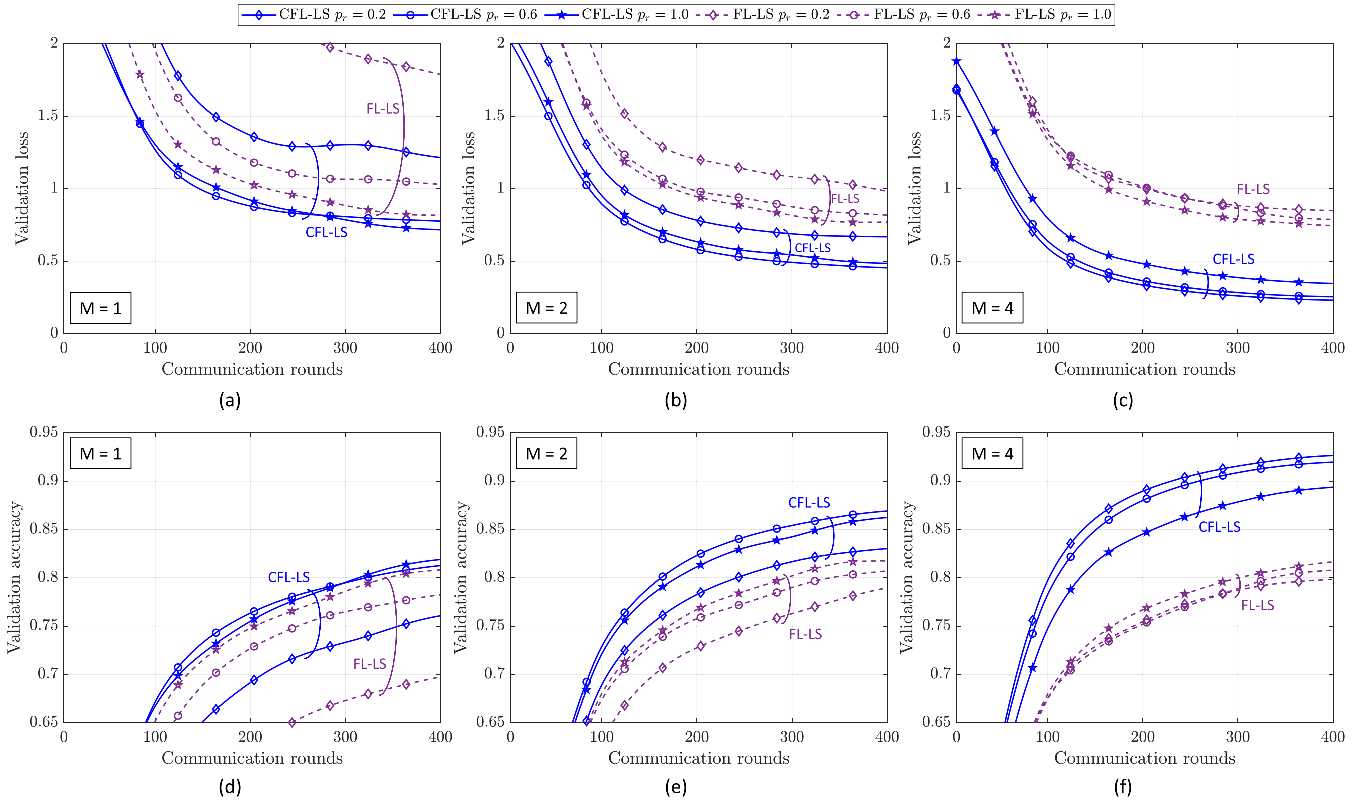


FIGURE 7. Analysis of the performance achieved by the proposed method and centralized FL under the same connectivity pattern. Results show the validation loss obtained for: (a) $M = 1$, (b) $M = 2$, and (c) $M = 4$. Similarly, the validation accuracy considers: (d) $M = 1$, (e) $M = 2$, and (f) $M = 4$.

hand, choosing layer selection policies that prioritize gradient information is beneficial as the number of parameters exchanged among the cooperating agents increases. Looking at the performances achieved when $M = 2$, the policy with $p_r = 0.6$ shows the best validation accuracy/loss, while, for $M = 4$, $p_r = 0.2$ should be preferred. These choices are especially important when considering 10% connectivity patterns as the difference among p_r values grows, whereas they are not so crucial for 50% and 90% connectivity cases apart for $M = 1$. Indeed, when $M = 4$ choosing $p_r = 1.0$, i.e. the full randomized selection policy, greatly impacts the overall performances compared to $p_r = 0.2$. Table 2 summarizes the validation loss/accuracy obtained by all methods at the end of the training process, highlighting that the proposed approach reaches nearly the same performances of FL and C-FL but with much lower communication overhead.

To visualize how the layer selection policy behaves with varying probability p_r, we report in Fig. 6 the percentage of times that each layer has been transmitted during the FL process considering $M = 1$ (Fig. 6a), $M = 2$ (Fig. 6b) and $M = 4$ (Fig. 6c). The layer numbering is the same as the one presented in Table 1, while the connectivity considered is the 50% one. From the analysis it can be noticed that for $M = 1$ and for $p_r = 0.2$, almost 80% of the times the 6-th layer is selected for transmission, which has also the second lowest number of parameters. As M increases, setting $p_r = 0.2$ provides a reasonable balance between the need of prioritizing the most informative layers/parameters before transmission and the requirement of sharing all the parameters of the model during the FL process. An intermediate case is presented by $p_r = 0.6$ where the 6-th layer is now transmitted almost half the times as opposed to 80% in $p_r = 0.2$. Finally, the policy with $p_r = 1$, being the most fair out of the three, shares all the layers in the same manner, as expected.

To conclude the analysis on the MNIST dataset, we compare the performances of CFL-LS with the vanilla FL policy based on the PS orchestration under the same connectivity conditions. For the latter method, we assume that the devices participating to the federation employ the same gradient sorting and randomized operations for selecting the layers to forward to the PS for aggregation. Therefore, in line with CFL-LS, the FL method is here referred to as Federated Learning with Layer Selection (FL-LS). Layer selection is implemented focusing on $p_r = \{0.2, 0.6, 1.0\}$. As far as connectivity in FL-LS is concerned, the PS randomly chooses the 50% of the devices for updating the global model at each training round, in line with the 50% connectivity scenario analyzed previously. Fig. 7 shows the validation loss (top) and validation accuracy (bottom) when $M = 1$ (Fig. 7a and Fig. 7d), $M = 2$ (Fig. 7b and Fig. 7e) and $M = 4$ (Fig. 7c and Fig. 7f). The numerical results show that the proposed method outperforms the vanilla FL scheme by a large margin on all cases. One major difference between FL-LS and CFL-LS resides in the optimized choice of p_r as the number of transmitted layers M increases. Indeed, the FL-LS policy is shown to be superior when choosing $p_r = 1$ regardless of how many layers can be shared across the network. On the other hand, CFL-LS requires a careful selection of p_r which needs to take into account also the current value of M, as discussed previously.

D. IMPACT OF QUANTIZATION ON LAYER SELECTION PERFORMANCE

Compression schemes can be applied to the model updates exchanged among neighbors to further preserve communication resources. To study its impact on learning performances, we apply here the compression strategy adopted in [33] to the CFL-LS method. The analysis evaluates the quantization effects with $b_{k,t} = \{8, 10\}$ bits on the validation loss/accuracy for $p_r = 0.2$ in the 50% connectivity case. The performances are also compared with the CFL-LS method without introducing any quantization with $b_{k,t} = 32$ bits (a typical number employed to encode ML model parameters in most deep learning frameworks).

Fig. 8 shows the validation loss (top) and validation accuracy (bottom) of the considered CFL-LS technique for $M = \{1, 2, 4\}$. The analysis shows that a sufficient number of bits should be devoted to encode the transmitted layers to not incur in performance degradation. This happens for all values of M even though the performance loss is marginal, especially for $M = 4$. Interestingly, a slight improvement on the final performances is observed when $b_{k,t} = 10$ bits, indicating that relying on uncompressed transmission schemes may provide models that generalize less for the considered learning task [51].

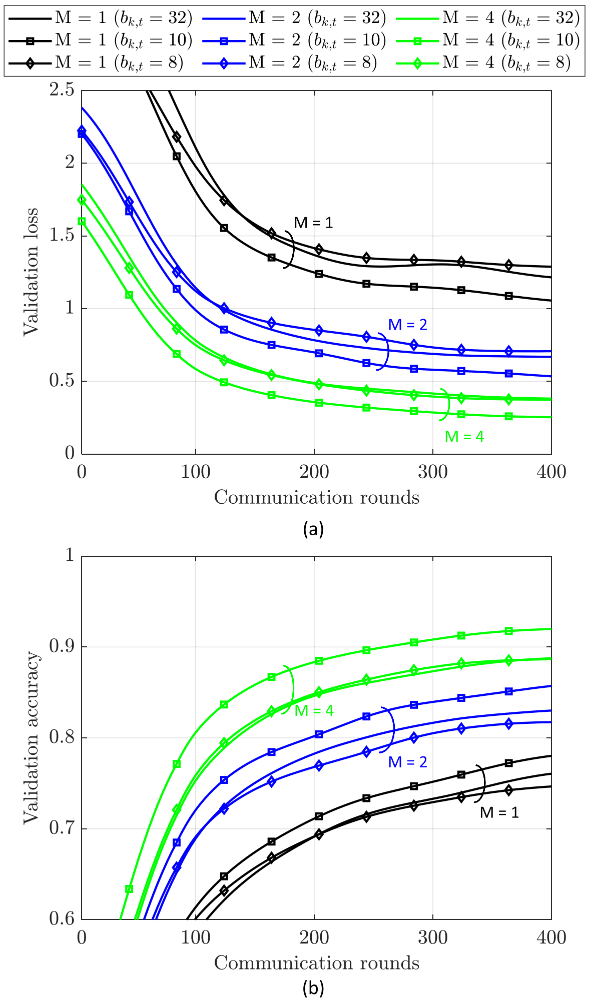


FIGURE 8. Analysis of the quantization impact on learning performance: (a) validation loss, (b) validation accuracy.

VI. COOPERATIVE SENSING IN VEHICULAR NETWORKS: A CASE STUDY

This section is dedicated to the assessment of the proposed methods over a more challenging vehicular sensing use case. We consider the cooperative sensing scenario in Fig. 9, where a number of connected vehicles use onboard lidar sensors to detect road users and/or relevant objects in the surroundings for automated driving services. The vehicles employ a DNN model that processes the Lidar point clouds and outputs the category of the detected road entity. The FL model is continuously trained in a cooperative manner by exchanging model updates through V2V sidelink communications.

In the considered scenario, the vehicles aggregate 10 Lidar sweeps to densify each point cloud data and process it through a local bounding box subsystem that provides object segmentation and position information, as depicted in the bottom part of Fig. 9. According to the bounding box position, extent and rotation, the point clouds that fall within the boxes are extracted and fed to the classifier (depicted by the classification subsystem in Fig. 9). The FL process acts only on the point cloud classification, while the bounding box regression is implemented locally at each vehicle.

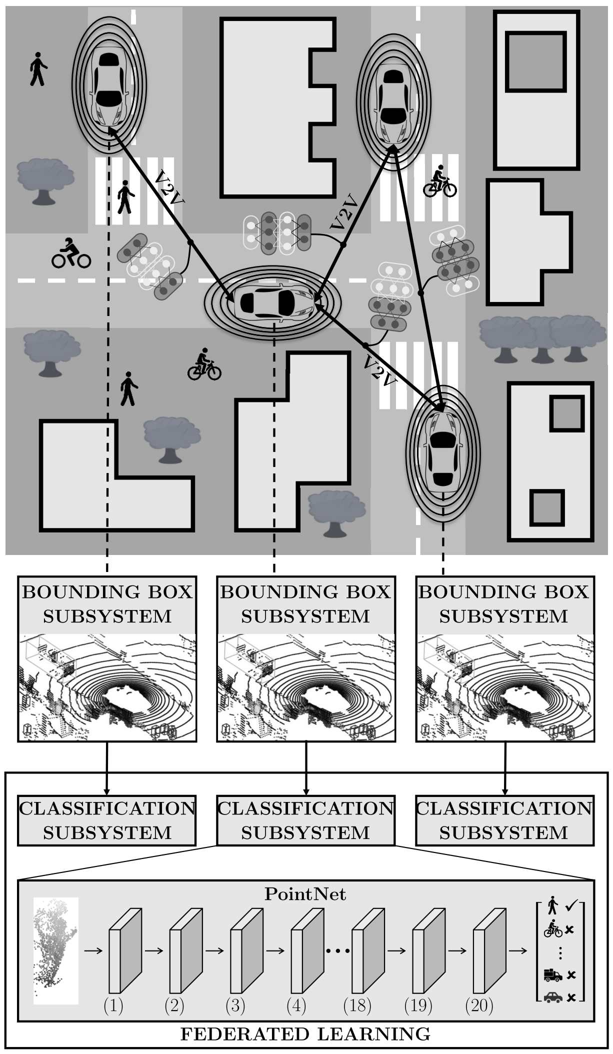


FIGURE 9. Vehicular case study: vehicles aim to optimize a ML model for road user classification from point cloud data by implementing a FL training scheme. Each vehicle implements a two-stage processing pipeline: i) bounding box regression and ii) PointNet-based classification. FL optimizes only the PointNet ML architecture.

As far as the decentralized FL training is concerned, vehicles collect point cloud data as they move in the environment and upon gathering enough samples they communicate a single time with a Road Side Unit (RSU) that is tasked to label the examples provided by vehicles. Vehicles belonging to different areas may communicate with different RSUs and provide rather different training categories that will reflect location-dependent properties of road users/objects. For example, vehicles moving on highways will gather little data regarding pedestrians compared to vehicles moving in urban environments. Once all vehicles acquire their corresponding training dataset, we propose to carry out a decentralized FL scheme to let the vehicles train on the overall collected data. Notice that the decentralization of the FL process allows to take some load off from the RSUs thus saving communication resources. Furthermore, it avoids communication among the RSUs which may come at an increased cost compared to V2V

TABLE 3. PointNet model adaptation.

Layer (number)	Filters/Neurons	Parameters	Size (Kb)
CL1 (1)	8	32	1.02
CL2 (2)	16	144	4.61
CL3 (3)	128	2176	69.63
FC1 (4)	64	8256	264.19
FC2 (5)	32	2080	66.56
FC3 (6)	9	297	9.50
CL4 (7)	8	32	1.02
CL5 (8)	8	72	2.30
CL6 (9)	8	72	2.30
CL7 (10)	16	144	4.61
CL8 (11)	128	2176	69.63
FC4 (12)	64	8256	264.19
FC5 (13)	32	2080	66.56
FC6 (14)	64	2112	67.58
CL9 (15)	8	72	2.30
CL10 (16)	16	144	4.61
CL11 (17)	128	2176	69.63
FC7 (18)	64	8256	264.19
FC8 (19)	32	2080	66.56
FC9 (20)	6	198	6.34

interactions and also be sporadic, intermittent and unavailable depending on the current load of the RSUs.

In what follows, Sec. VI-A presents the adopted ML model and the related datasets. Sec. VI-B presents the performances of the proposed method, again compared to several baselines: the vanilla FL tool, the conventional fully decentralized FL technique, and the DML approach. All baselines are implemented without introducing compression strategies. Sec. VI-C studies the CFL-LS algorithm considering link unavailability events. Finally, Sec. VI-D thoroughly analyzes the communication efficiency vs model quality tradeoff for different layer selection strategies.

A. MODEL AND FEDERATED DATASETS

The DNN model here employed is PointNet [52], largely used for 3D shape recognition and classification from point cloud data. The architecture is adapted from [28], with parameters for each layer defined in Table 3. The modified architecture relies on $L = 20$ federated layers with 40855 potential trainable parameters that can be selected for transmission on each FL round. Note that batch normalization layers are updated opportunistically based on the available local data at each vehicle.

The proposed CFL-LS tool is assessed using the publicly available nuScenes Lidar dataset [53] for autonomous driving. These real-world data have been collected by a fully-equipped vehicle in challenging situations, such as diverse weather/lighting conditions as well as traffic densities. The training dataset is generated as in [28] and contains 9000 training example pairs $(\mathbf{x}_h, \mathbf{y}_h)$ where $\mathbf{x}_h$ is the point cloud and $\mathbf{y}_h$ is the corresponding road category chosen among 6 classes. Similarly, the validation set contains 2400 evenly distributed Lidar sets across the 6 classes and is used for performance assessment (accuracy/loss). To comply with the PointNet ML model, which accepts a fixed number of points, $\mathbf{x}_h$ is upsampled/downsampled to contain exactly 2048 points. Furthermore, we normalize $\mathbf{x}_h$ such that it is contained into a unit area sphere. The

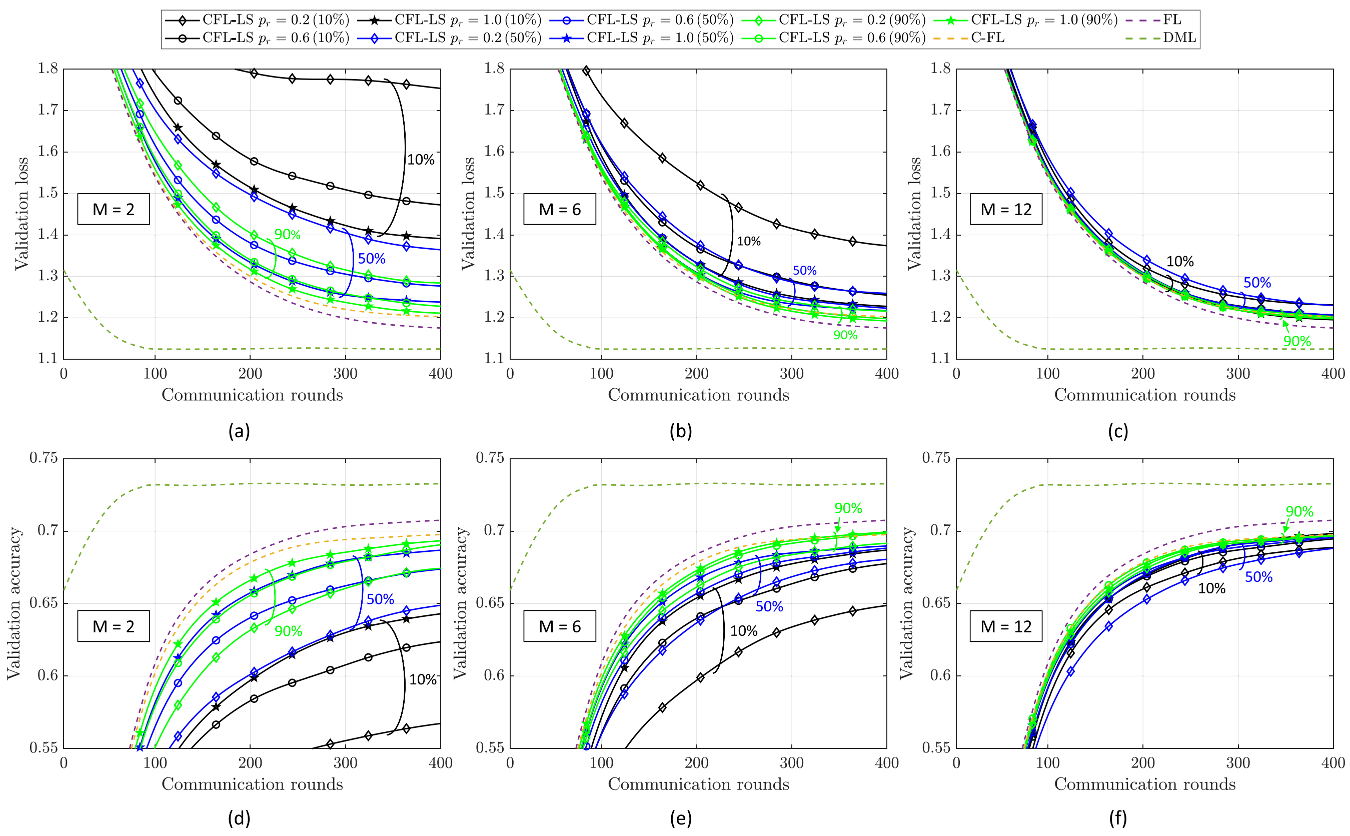


FIGURE 10. Performance analysis of the CFL-LS method in terms of validation loss (top) and accuracy (bottom) for number of shared layers (a,d) $M = 2$, (b,e) $M = 6$ and (c,f) $M = 12$. Percentages indicate which connectivity pattern is considered among 10%, 50% and 90% cases.

overall training process is simulated in a virtual environment, where $N = 10$ vehicles are deployed. The vehicles' dynamics have not been taken into account as the main goal of the paper is to study the proposed layer-wise compression operators on the FL performance. Nevertheless, interested readers may refer to [28] for insights on how mobility affects the learning performances. At the start of the training, the overall data is partitioned across the vehicles participating to the distributed learning process. In particular, each vehicle holds 200 examples evenly partitioned into 5 of the 6 road categories to simulate non-iid local datasets.

In the following, we evaluate the CFL-LS approach by varying the constraint on the spectral efficiency defined in (7) with $P_U = 0$, allowing the sharing of $M = 2$ up to $M = 12$ layers. Given a frame slot duration $T_F = 45$ ms and $b_{k,t} = 32$ bits, the corresponding bandwidth B_W ranges from 1.45 up to 8.7 MHz. The analysis focuses also on the assessment of the CFL-LS method by changing the threshold probabilities p_r and the connectivity patterns (from sparse to dense), defined in Sec. V-A. Unless stated otherwise, results are averaged over 5 independent runs and over all vehicles.

B. MODEL QUALITY ASSESSMENT

Similarly as done in Sec. IV, we evaluate the performances obtained by CFL-LS as compared to CFL, FL and DML training schemes. Fig. 10 shows the validation loss (top) and validation accuracy (bottom) for $M = 2$ (Fig. 10a and Fig. 10d), $M = 6$ (Fig. 10b and Fig. 10e) and $M = 12$ (Fig. 10c and Fig. 10f). The numerical results confirm the findings of the analysis in Fig. 5. Carefully selecting the probability threshold p_r taking into account M is beneficial for balancing the communication resources and the model quality. For example, choosing $p_r = 0.6$ when $M = 2$ over $p_r = 0.2$ for $M = 6$ allows to substantially reduce the number of transmitted data without introducing accuracy penalties. Opting for $p_r = 1$ when $M = 6$ rather than $p_r = 0.6$ for $M = 12$ heavily reduces the communication footprint without significant accuracy drops. CFL-LS approaches the performance of conventional CFL and FL tools but with a much lower communication burden as shown for $p_r = 1$ and $M = \{6, 12\}$. On the other hand, convergence rates shown by CFL and FL are now slightly superior compared to CFL-LS schemes.

We recall that the probability threshold p_r rules the fairness in sharing the layers among the devices. For low number of shared layers M (i.e., low communication footprint per round) randomized selection policies with large p_r, are recommended. Indeed, $p_r = 1$ should be selected for $M = \{2, 6\}$ for achieving the highest performances. On the other hand, $p_r = 0.6$ is favorable against $p_r = \{0.2, 1.0\}$ when $M = 12$. The choice of p_r may also be tuned taking into account the model depth and the distribution of the trainable

TABLE 4. Comparison of the validation accuracy and validation loss obtained over the Lidar dataset for all methods considered.

Layers	DML	FL	C-FL	CFL-LS								
				p_r (10 % connectivity)			p_r (50 % connectivity)			p_r (90 % connectivity)		
				0.2	0.6	1.0	0.2	0.6	1.0	0.2	0.6	1.0
$M = 2$ (val. loss)	**1.124**	**1.175**	**1.202**	1.754	1.473	**1.391**	1.364	1.277	**1.238**	1.284	1.228	**1.211**
$M = 6$ (val. loss)	**1.124**	**1.175**	**1.202**	1.374	1.255	**1.228**	1.259	1.223	**1.218**	1.216	1.198	**1.193**
$M = 12$ (val. loss)	**1.124**	**1.175**	**1.202**	1.231	**1.195**	1.206	1.229	**1.206**	1.207	1.203	**1.197**	1.202
$M = 2$ (val. accuracy)	**0.732**	**0.707**	**0.697**	0.567	0.623	**0.642**	0.648	0.673	**0.686**	0.674	0.690	**0.693**
$M = 6$ (val. accuracy)	**0.732**	**0.707**	**0.697**	0.648	0.677	**0.686**	0.680	0.688	**0.689**	0.691	0.698	**0.699**
$M = 12$ (val. accuracy)	**0.732**	**0.707**	**0.697**	0.688	**0.698**	0.694	0.688	**0.696**	0.695	0.696	**0.698**	0.697

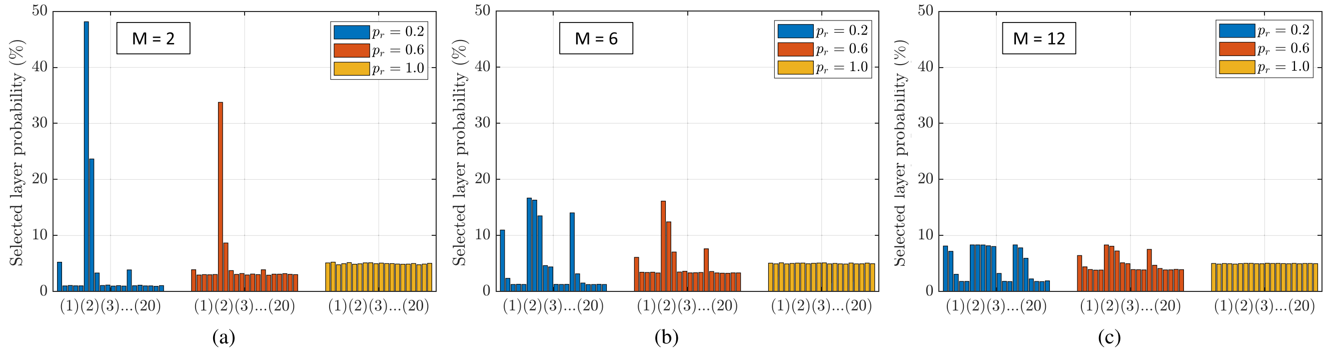


FIGURE 11. Probability of selected layers from the PointNet model, with (a) $M = 2$, (b) $M = 6$ and (c) $M = 12$. Layers (x) for x = 1, …, 20 are defined in Table 3.

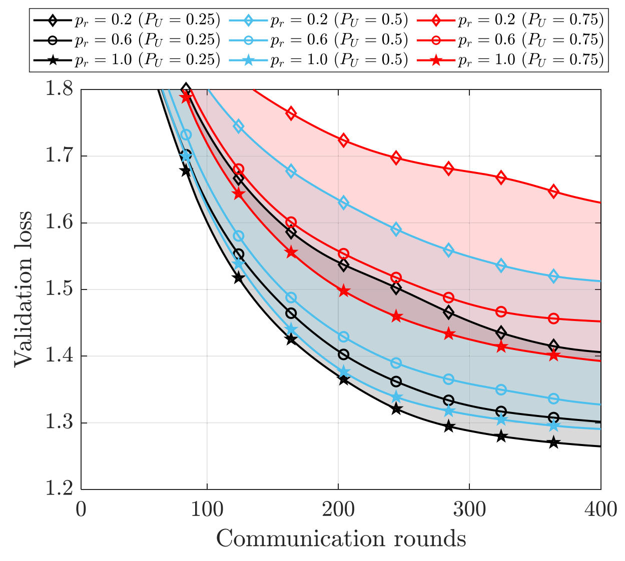
parameters for each layer. The considered architecture in this case study shows many layers having very few parameters, i.e., from 32 up to 144 for 35% of the layers, indicating that $p_r = 0.2$ may be a superior choice for larger values of M. Furthermore, integrating batch normalization layers in the architecture might impact the learning performances, especially for non-iid data distributions across vehicles participating in the federated process, as pointed out in [54]. The obtained results are summarized in Table 4 showing that the proposed selection strategies reach nearly the same performances of other baselines.

Similarly as done in Sec. V-C, Fig. 11 reports the probabilities of the chosen PointNet layers during the CFL-LS process, considering the probability thresholds $p_r = 0.2$, $p_r = 0.6$ and $p_r = 1.0$, respectively, and $M = 2$ (Fig. 11a), $M = 6$ (Fig. 11b) and $M = 12$ (Fig. 11c). Results are also presented for the 50% connectivity scenario. For $p_r = 0.2$, the policy favors the selection of the layers labeled from 6 to 8 in Table 3, and positioned roughly at the middle of the PointNet structure, namely between the first and the second transformation mini-networks [52]. As M increases, the selected model parameters still belong to layers close to layers 6, 7 and 8. Initial and final layers are also chosen in some cases. This is opposed to the previous results for MNIST processing, where the policy prioritized the parameters close to the DNN output. Interestingly, in both cases the proposed selection strategy tends to prioritize the layers possessing fewer parameters, suggesting also that layers containing many trainable parameters can be shared less frequently.

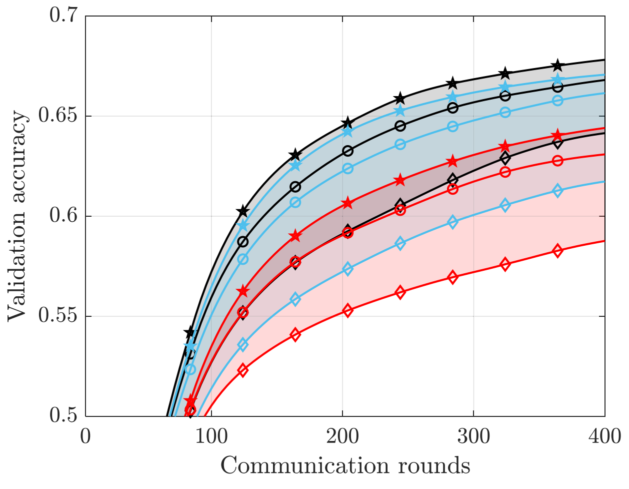
C. IMPACT OF COMMUNICATION IMPAIRMENTS

Poor or intermittent communications may heavily impact the final quality of the trained models. In the following, we thus study the robustness of the CFL-LS method against link outage events. We set up a communication framework that allows us to simulate connection drops among vehicles according to a pre-defined probability, namely P_U in (6), and on a per-layer basis, meaning that when a link is unavailable the layer(s) cannot be transmitted as no connection exists. Link unavailability events are assumed to be independent and identically distributed (iid) across all participating vehicles to the FL process and over all model layers transmitted.

In Fig. 12 the CFL-LS performances are assessed for $M = 2$ and 50% connectivity. We evaluate 3 highly challenging scenarios representing extremely poor link availability in the network, ranging from $P_U = 0.25$ up to $P_U = 0.75$. This is done for evaluating the proposed approach in extremely challenging communication conditions, showing how CFL-LS responds to such detrimental effects. The figure reports the validation loss (Fig. 12a) and validation accuracy (Fig. 12b) considering the three P_U probabilities separately, namely for $P_U = \{0.25, 0.50, 0.75\}$. Increasing P_U results in worse performances, as expected. Thus, the selection of p_r becomes extremely important when considering a large number of unavailable links in the network: $p_r = 1.0$ should be always chosen to overcome such events as it provides the highest performances when compared to all other choices. Furthermore, the accuracy drop between $P_U = 0.25$ and $P_U = 0.5$ when $p_r = 1.0$ is relatively small when compared against $p_r = 0.2$ and $p_r = 0.6$, indicating



(a)



(b)

FIGURE 12. Analysis of the robustness of the CFL-LS approach over different P_U values: (a) validation loss, (b) validation accuracy.

that choosing full randomized layer selection schemes may alleviate the communication impairments effects on learning performance. Unfortunately, when the number of available links is scarce, i.e., $P_U = 0.75$, also $p_r = 1.0$ suffers large performance drops, suggesting that special countermeasures should be put in place or allowing the FL process to run for more learning rounds.

D. COMMUNICATION EFFICIENCY VS MODEL QUALITY TRADE-OFF

To conclude the case study analysis, we now quantify the communication resources, namely the *communication footprint*, needed by the CFL-LS policy to reach a target validation loss value. The goal is to evaluate the tradeoff between communication-efficiency and model quality. The communication footprint corresponds to the amount of data exchanged over the network during an assigned number of FL rounds. Footprint results are shown considering two different wall clock times that comprise up to 100 and 400 FL communication rounds, respectively. In line with [55], it is assumed that the selected model parameters can be shared among neighbors using broadcast messages without requiring each vehicle to forward one copy of the shared layers for every neighbor.

Fig. 13 depicts the results of the validation procedure considering sparse to dense connectivity scenarios, namely 10% (Fig. 13a), 50% (Fig. 13b), and 90% (Fig. 13c) connected vehicles, respectively. Each point in the scatter plot represents the communication footprint and the corresponding validation loss obtained by a vehicle participating in the FL process. Each marker encodes the information regarding the probability p_r used by the layer selection policy: diamond, circle and star symbols indicate $p_r = 0.2$, $p_r = 0.6$ and $p_r = 1.0$, respectively. Colors refer to different choices for the number of transmitted layers M, with black, blue and green corresponding to $M = 2$, $M = 6$ and $M = 12$, respectively. For each vehicle participating in the FL process, we measure the validation loss observed after 100 and 400 training rounds and compute, for each case, how many parameters, in MB, have been exchanged until that point. The loss values obtained are also averaged over 5 independent runs.

Focusing on the overall performances, the numerical results show that $p_r = 0.2$ gives the best results in terms of communication efficiency for all the considered connectivity patterns and wall clock times, while $p_r = 1.0$ is the least efficient one. On the other hand, $p_r = 0.2$ is also the least performing in terms of model quality for all cases considered in the figures. Trade-offs between communication efficiency and model quality need to be considered depending on the available network resources, the desired accuracy and training latency. For example, $p_r = 0.2$ exhibits the lowest communication footprint when compared to all other methods, making it the most favorable choice provided that the target loss $\mathcal{L}_i(\mathbf{W})$ is below 1.8. On the other hand, for $\mathcal{L}_i(\mathbf{W}) = 1.2$, $p_r = 0.6$ should be preferred as exhibiting the lowest communication footprint among all other methods and validation loss in line with $p_r = 1.0$. Analyzing the results for different values of the transmitted layers M, it can be noticed that $M = 12$ should be avoided as responsible for a high communication footprint, compared with other setups. Instead, sending $M = 6$ layers per round provides the best tradeoff as reducing the required network resources compared with $M = 12$ in exchange for some (marginal) degradation of model accuracy. Finally, sending $M = 2$ layers further reduces the required footprint, however much lower accuracy, i.e., a 5%-8% increase in validation loss, is observed.

Concerning the connectivity patterns, results indicate that the number of sidelink connections has a significant impact on the consensus process. In particular, the validation loss across all vehicles is less dispersed as the number of connections increases. Sparse connectivity makes consensus converge slowly and is responsible for large variations of the validation loss across the vehicles. This effect is more evident after 100 training rounds. Whereas, such variability

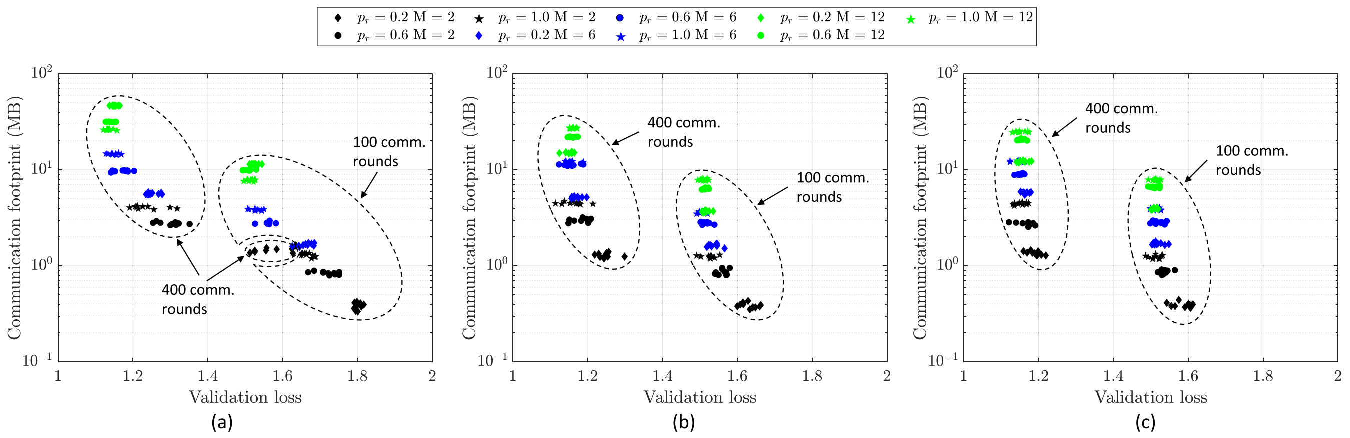


FIGURE 13. **Analysis of the communication footprint vs model quality tradeoff for different degree of connectivity: (a) 10%, (b) 50% and (c) 90%. Results obtained after 100 and 400 communication rounds are also highlighted (dashed ellipses).**

reduces considering more dense connectivity scenarios (50% and 90% patterns). Optimizing communication efficiency and model quality in sparse connectivity scenarios is thus fundamental since choosing an inappropriate value of M and p_r might lead to high validation loss. On the other hand, dense sidelink communications allow the vehicles to keep the number of exchanged layers M as low as possible, thus maximizing the communication efficiency: for the considered study, setting $p_r = 0.2$ and $M = 2$ provides a reasonable tradeoff between communication footprint and model accuracy.

VII. CONCLUSION

In this paper, we analyzed the communication efficiency of fully decentralized FL setups underpinned by consensus tools. We designed a novel communication-efficient FL framework that enables the agents to self-organize into a distributed training platform while optimizing the network resources used for the learning process. The proposed CFL-LS method employs a *layer optimizer* that selects the NN layers to be shared by sorting them according to their contribution to the model quality, measured by the normalized squared gradient of the local loss. The layer selection policy is integrated with a fairness scheme that selects randomly the layers in the ML architecture so as to favor a balanced selection of the ML model parameters and optimize the performance.

The proposed layer selection optimizer is firstly analyzed to study its impact on the consensus process. The analysis shows that the proposed solution does not reach average consensus. Nevertheless, the convergence rate provided by the FL method is far superior when compared to a coordinator-based strategy integrating the same layer selection policy that instead achieves average consensus. Then, the communication-efficient FL technique is assessed on the benchmark MNIST as well as on the more challenging nuScenes dataset, targeting a cooperative vehicular sensing use case. The proposed CFL-LS layer selection policy has been validated with a PointNet-compliant DNN architecture composed by 40 trainable layers. This is used to reliably and precisely recognize road users from Lidar point clouds. Latency, accuracy and communication-efficient trade-offs have been extensively analyzed to evaluate the performance. Results indicate that the proposed layer selection policy reduces significantly the communication overhead needed during the training process, in exchange for negligible performance loss compared to classical centralized (FL) and decentralized (CFL) policies. The analysis also shows how balancing gradient sorting operations and randomization for layer selection helps to reduce the communication burden, without penalizing accuracy or convergence rates. More specifically, the main takeaways can be summarized as: i) randomized selection policies ($p_r = 1$) should be preferred when communication resources are scarce; ii) gradient-based selection ($p_r \in [0, 0.2]$) should be instead chosen when communication resources are not critical (10-100 MB); iii) prioritizing the exchange of layers possessing *few* trainable parameters is beneficial for improving communication-efficiency. Finally, experimental tests show that the proposed approach is suitable for integration with device scheduling functions, as well as network quantization schemes.

APPENDIX A

Let us focus on the consensus equation (23) and assume that $\mathbf{L}_{FL}$ satisfies the following conditions: $\mathbf{1}^{\mathrm{T}}\mathbf{L}_{FL} = \mathbf{0}$ and $\mathbf{L}_{FL}\mathbf{1} = \mathbf{0}$ so that the Perron matrix $\mathbf{P}_{\bar{\mathbf{a}}(q)}$ is doubly stochastic. We apply the eigenvalue decomposition to $\mathbf{P}_{\bar{\mathbf{a}}(q)}$ and obtain $\mathbf{P}_{\bar{\mathbf{a}}(q)} = \mathbf{U}\boldsymbol{\Sigma}\mathbf{V}^{\mathrm{T}}$, where $\mathbf{U} = [\mathbf{u}_1 \cdots \mathbf{u}_{NL}]$ and $\mathbf{V} = [\mathbf{v}_1 \cdots \mathbf{v}_{NL}]$ are the left and right eigenvectors, respectively, while $\boldsymbol{\Sigma} = \mathrm{diag}[\lambda_1, \ldots, \lambda_{LN}]$ is a diagonal matrix containing the eigenvalues in non-descending order. Recalling that $\mathbf{1}^{\mathrm{T}}\mathbf{L}_{FL} = \mathbf{0}$ and $\mathbf{L}_{FL}\mathbf{1} = \mathbf{0}$ and $\mathbf{L}_{FL}$ is block-partitioned into L blocks, it follows that $\mathbf{L}_{FL}$ has L trivial eigenvalues with value 0. This implies that the first L eigenvalues of $\mathbf{P}_{\bar{\mathbf{a}}(q)}$ are $\lambda_1 = \ldots = \lambda_L = 1$ while $\mathbf{U}_0 = \mathbf{V}_0 = \mathbf{1} \otimes \mathbf{I}_L$ are the associated left/right eigenvector matrices as satisfying the following conditions

$$\mathbf{V}_0^{\mathrm{T}}\mathbf{L}_{FL} = \mathbf{1}^{\mathrm{T}}\mathbf{L}_{FL} \otimes \mathbf{I}_L = \mathbf{0}_{L\times LN} = \mathbf{0}_{L\times L}\mathbf{V}_0^{\mathrm{T}} \tag{27}$$

$$\mathbf{L}_{FL}\mathbf{U}_0 = \mathbf{L}_{FL}\mathbf{1} \otimes \mathbf{I}_L = \mathbf{0}_{NL\times L} = \mathbf{U}_0\mathbf{0}_{L\times L} \tag{28}$$

$$\mathbf{V}_0^{\mathrm{T}}\mathbf{U}_0 = \mathbf{I}_L. \tag{29}$$

Combining the above results, (23) can be rewritten as

$$\mathcal{W}_{t+1} = \mathbf{U}_0\mathbf{\Sigma}_0\mathbf{V}_0^{\mathrm{T}}\mathcal{W}_{t_0} + \textstyle\sum_{n=L+1}^{LN} \lambda_n \mathbf{u}_n \mathbf{v}_n^{\mathrm{T}} \mathcal{W}_{t_0} \tag{30}$$

where $\mathbf{u}_n$ and $\mathbf{v}_n$ are the left and right eigenvectors associated to the eigenvalues λ_n with $n = L+1, \ldots, LN$. Convergence to the average initial values contained in $\mathcal{W}_{t_0}$ is thus obtained when the summation of (30) approaches zero, i.e., for $|\lambda_n| = |1 - \varepsilon\mu_n| \leq 1$, or equivalently

$$0 \leq \varepsilon \leq 2/\mu_{max}(\mathbf{L}_{FL}), \tag{31}$$

where $\mu_{max}(.)$ denotes the largest eigenvalue of $\mathbf{L}_{FL}$. By applying the Gershgorin theorem, we obtain:

$$\begin{aligned} |\mu_n(\mathbf{L}_{FL}) - [\mathbf{L}_{FL}]_{ii}| &\leq \sum_{j\neq i}[\mathbf{L}_{FL}]_{ij} = d_i \\ |\mu_n(\mathbf{L}_{FL}) - d_i| &\leq d_i \\ 0 \leq \mu_n(\mathbf{L}_{FL}) &\leq 2\, d_i. \end{aligned} \tag{32}$$

Therefore, convergence is guaranteed for:

$$0 \leq \varepsilon \leq 1/d_{\max}, \tag{33}$$

where d_{max} is the maximum connectivity degree. In the considered case, d_{max} is the maximum number of times a layer is chosen by all devices.

REFERENCES

[1] R. Olfati-Saber, J. A. Fax, and R. M. Murray, "Consensus and cooperation in networked multi-agent systems," *Proc. IEEE*, vol. 95, no. 1, pp. 215–233, Jan. 2007.

[2] S. Kar and J. M. F. Moura, "Consensus + innovations distributed inference over networks: Cooperation and sensing in networked systems," *IEEE Signal Process. Mag.*, vol. 30, no. 3, pp. 99–109, May 2013.

[3] J. B. Predd, S. R. Kulkarni, and H. V. Poor, "Distributed learning in wireless sensor networks," *IEEE Signal Process. Mag.*, vol. 23, no. 4, pp. 56–69, Jul. 2006.

[4] G. Soatti, M. Nicoli, S. Savazzi, and U. Spagnolini, "Consensus-based algorithms for distributed network-state estimation and localization," *IEEE Trans. Signal Inf. Process. Netw.*, vol. 3, no. 2, pp. 430–444, Mar. 2017.

[5] Z. Li, B. Liu, and Z. Ding, "Consensus-based cooperative algorithms for training over distributed data sets using stochastic gradients," *IEEE Trans. Neural Netw. Learn. Syst.*, vol. 33, no. 10, pp. 5579–5589, Oct. 2022.

[6] K. I. Tsianos, S. Lawlor, and M. G. Rabbat, "Consensus-based distributed optimization: Practical issues and applications in large-scale machine learning," in *Proc. 50th Annu. Allerton Conf. Commun., Control, Comput. (Allerton)*, Oct. 2012, pp. 1543–1550.

[7] R. Olfati-Saber and R. M. Murray, "Consensus problems in networks of agents with switching topology and time-delays," *IEEE Trans. Autom. Control*, vol. 49, no. 9, pp. 1520–1533, Sep. 2004.

[8] S.-Y. Tu and A. H. Sayed, "Distributed decision-making over adaptive networks," *IEEE Trans. Signal Process.*, vol. 62, no. 5, pp. 1054–1069, Mar. 2014.

[9] S. Li, G. Oikonomou, T. Tryfonas, T. M. Chen, and L. D. Xu, "A distributed consensus algorithm for decision making in service-oriented Internet of Things," *IEEE Trans. Ind. Informat.*, vol. 10, no. 2, pp. 1461–1468, May 2014.

[10] W. Saad, M. Bennis, and M. Chen, "A vision of 6G wireless systems: Applications, trends, technologies, and open research problems," *IEEE Netw.*, vol. 34, no. 3, pp. 134–142, May 2020.

[11] H. Tataria, M. Shafi, A. F. Molisch, M. Dohler, H. Sjöland, and F. Tufvesson, "6G wireless systems: Vision, requirements, challenges, insights, and opportunities," *Proc. IEEE*, vol. 109, no. 7, pp. 1166–1199, Jul. 2021.

[12] L. U. Khan, I. Yaqoob, M. Imran, Z. Han, and C. S. Hong, "6G wireless systems: A vision, architectural elements, and future directions," *IEEE Access*, vol. 8, pp. 147029–147044, 2020.

[13] S.-Y. Lien, D.-J. Deng, C.-C. Lin, H.-L. Tsai, T. Chen, C. Guo, and S.-M. Cheng, "3GPP NR sidelink transmissions toward 5G V2X," *IEEE Access*, vol. 8, pp. 35368–35382, 2020.

[14] M. Harounabadi, D. M. Soleymani, S. Bhadauria, M. Leyh, and E. Roth-Mandutz, "V2X in 3GPP standardization: NR sidelink in release-16 and beyond," *IEEE Commun. Standards Mag.*, vol. 5, no. 1, pp. 12–21, Mar. 2021.

[15] 5G ACIA. *5G Alliance for Connected Industries and Automation*. Accessed: Dec. 2021. [Online]. Available: https://5g-acia.org/

[16] *ETSI, LTE;5G; Overall Description of Radio Access Network (RAN) Aspects for Vehicle-to-Everything (V2X) Based on LTE and NR*, document ETSI TR 137 985, Tech. Rep. 137 985, Version 16.0.0, European Telecommunications Standards Institute (ETSI), Dec. 2020.

[17] *3rd Generation Partnership Project; Technical Specification Group Services and System Aspects; Architecture Enhancements for 5G System (5GS) to Support Vehicle-to-Everything (V2X) Services (Release 17)*, Standard 3GPP TS 23.287 V17.1.0, 2021.

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

[19] B. McMahan, E. Moore, D. Ramage, S. Hampson, and B. A. Y. Arcas, "Communication-efficient learning of deep networks from decentralized data," in *Proc. 20th Int. Conf. Artif. Intell. Statist.*, vol. 54, 2017, pp. 1273–1282.

[20] P. Kairouz, "Advances and open problems in federated learning," *Found. Trends Mach. Learn.*, vol. 14, nos. 1–2, pp. 1–210, Jun. 2021, doi: 10.1561/2200000083.

[21] 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.

[22] O. A. Wahab, A. Mourad, H. Otrok, and T. Taleb, "Federated machine learning: Survey, multi-level classification, desirable criteria and future directions in communication and networking systems," *IEEE Commun. Surveys Tuts.*, vol. 23, no. 2, pp. 1342–1397, 2021.

[23] J. Dean, "Large scale distributed deep networks," in *Proc. 25th NIPS*, vol. 1. Red Hook, NY, USA: Curran Associates, Dec. 2012, pp. 1223–1231.

[24] S. Savazzi, M. Nicoli, and V. Rampa, "Federated learning with cooperating devices: A consensus approach for massive IoT networks," *IEEE Internet Things J.*, vol. 7, no. 5, pp. 4641–4654, May 2020.

[25] S. Savazzi, M. Nicoli, M. Bennis, S. Kianoush, and L. Barbieri, "Opportunities of federated learning in connected, cooperative, and automated industrial systems," *IEEE Commun. Mag.*, vol. 59, no. 2, pp. 16–21, Feb. 2021.

[26] H. Xing, O. Simeone, and S. Bi, "Federated learning over wireless device-to-device networks: Algorithms and convergence analysis," *IEEE J. Sel. Areas Commun.*, vol. 39, no. 12, pp. 3723–3741, 2021, doi: 10.1109/JSAC.2021.3118400.

[27] L. Barbieri, S. Savazzi, and M. Nicoli, "Decentralized federated learning for road user classification in enhanced V2X networks," in *Proc. IEEE Int. Conf. Commun. Workshops (ICC Workshops)*, Jun. 2021, pp. 1–6.

[28] L. Barbieri, S. Savazzi, M. Brambilla, and M. Nicoli, "Decentralized federated learning for extended sensing in 6G connected vehicles," *Veh. Commun.*, vol. 33, Jan. 2022, Art. no. 100396.

[29] M. Chen, "Distributed learning in wireless networks: Recent progress and future challenges," *IEEE J. Sel. Areas Commun.*, vol. 39, no. 12, pp. 3579–3605, Dec. 2021.

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

[31] 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, Oct. 2020.

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

[33] A. Reisizadeh, A. Mokhtari, H. Hassani, A. Jadbabaie, and R. Pedarsani, "FedPAQ: A communication-efficient federated learning method with periodic averaging and quantization," in *Proc. 23rd Int. Conf. Artif. Intell. Statist.*, vol. 108, 2020, pp. 2021–2031.

[34] J. Konečný, H. Brendan McMahan, F. X. Yu, P. Richtárik, A. Theertha Suresh, and D. Bacon, "Federated learning: Strategies for improving communication efficiency," 2016, *arXiv:1610.05492*.

[35] A. Elgabli, J. Park, S. Ahmed, and M. Bennis, "L-FGADMM: Layer-wise federated group ADMM for communication efficient decentralized deep learning," in *Proc. IEEE Wireless Commun. Netw. Conf. (WCNC)*, May 2020, pp. 1–6.

[36] C. Hu, J. Jiang, and Z. Wang, "Decentralized federated learning: A segmented gossip approach," 2019, *arXiv:1908.07782*.

[37] J. Jiang, L. Hu, C. Hu, J. Liu, and Z. Wang, "BACombo—Bandwidth-aware decentralized federated learning," *Electronics*, vol. 9, no. 3, p. 440, Mar. 2020.

[38] C. Li, G. Li, and P. K. Varshney, "Decentralized federated learning via mutual knowledge transfer," *IEEE Internet Things J.*, vol. 9, no. 2, pp. 1136–1147, Jan. 2022.

[39] S. Lu, Y. Zhang, and Y. Wang, "Decentralized federated learning for electronic health records," in *Proc. 54th Annu. Conf. Inf. Sci. Syst. (CISS)*, Mar. 2020, pp. 1–5.

[40] S. Shi, Z. Tang, Q. Wang, K. Zhao, and X. Chu, "Layer-wise adaptive gradient sparsification for distributed deep learning with convergence guarantees," in *Proc. ECAI*, 2020, pp. 1–8.

[41] Z. Zhang, Y. Hu, and Q. Ye, "LR-SGD: Layer-based random SGD for distributed deep learning," in *Proc. 8th Int. Conf. Comput. Artif. Intell.*, New York, NY, USA: Association for Computing Machinery, Mar. 2022, pp. 6–11.

[42] S. Shi, Q. Wang, X. Chu, B. Li, Y. Qin, R. Liu, and X. Zhao, "Communication-efficient distributed deep learning with merged gradient sparsification on GPUs," in *Proc. IEEE INFOCOM - IEEE Conf. Comput. Commun.*, Jul. 2020, pp. 406–415.

[43] Y. Hong and P. Han, "LSDDL: Layer-wise sparsification for distributed deep learning," *Big Data Res.*, vol. 26, Nov. 2021, Art. no. 100272.

[44] L. Deng, "The MNIST database of handwritten digit images for machine learning research [best of the web]," *IEEE Signal Process. Mag.*, vol. 29, no. 6, pp. 141–142, Nov. 2012.

[45] S. Savazzi, S. Kianoush, V. Rampa, and M. Bennis, "A joint decentralized federated learning and communications framework for industrial networks," in *Proc. IEEE 25th Int. Workshop Comput. Aided Model. Design Commun. Links Netw. (CAMAD)*, Sep. 2020, pp. 1–7.

[46] D. Alistarh, D. Grubic, J. Z. Li, R. Tomioka, and M. Vojnovic, "Qsgd: Communication-efficient sgd via gradient quantization and encoding," in *Proc. 31st Int. Conf. Neural Inf. Process. Syst.*, 2017, pp. 1707–1718.

[47] D. P. Kingma and J. Ba, "Adam: A method for stochastic optimization," 2014, *arXiv:1412.6980*.

[48] A. Jadbabaie, J. Lin, and A. S. Morse, "Coordination of groups of mobile autonomous agents using nearest neighbor rules," *IEEE Trans. Autom. Control*, vol. 48, no. 6, pp. 988–1001, Jun. 2003.

[49] J. Wolfowitz, "Products of indecomposable, aperiodic, stochastic matrices," *Proc. Amer. Math. Soc.*, vol. 14, no. 5, pp. 733–737, May 1963.

[50] A. Tahbaz-Salehi and A. Jadbabaie, "A necessary and sufficient condition for consensus over random networks," *IEEE Trans. Autom. Control*, vol. 53, no. 3, pp. 791–795, Apr. 2008.

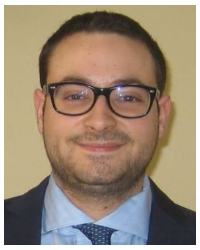
[51] A. Oland and B. Raj, "Reducing communication overhead in distributed learning by an order of magnitude (almost)," in *Proc. IEEE Int. Conf. Acoust., Speech Signal Process. (ICASSP)*, Apr. 2015, pp. 2219–2223.

[52] R. Q. Charles, H. Su, M. Kaichun, and L. J. Guibas, "PointNet: Deep learning on point sets for 3D classification and segmentation," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jul. 2017, pp. 77–85.

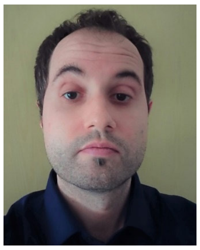
[53] H. Caesar, V. Bankiti, A. H. Lang, S. Vora, V. E. Liong, Q. Xu, A. Krishnan, Y. Pan, G. Baldan, and O. Beijbom, "NuScenes: A multimodal dataset for autonomous driving," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2020, pp. 11618–11628.

[54] X. Li, M. Jiang, X. Zhang, M. Kamp, and Q. Dou, "FedBN: Federated learning on non-IID features via local batch normalization," in *Proc. 9th Int. Conf. Learn. Represent., (ICLR)*, 2021, pp. 1–26.

[55] M. H. C. Garcia, A. Molina-Galan, M. Boban, J. Gozalvez, B. Coll-Perales, T. Sahin, and A. Kousaridas, "A tutorial on 5G nr V2X communications," *IEEE Commun. Surveys Tuts.*, vol. 23, no. 3, pp. 1972–2026, 3rd Quart., 2021.



LUCA BARBIERI (Graduate Student Member, IEEE) received the B.Sc. and M.Sc. degrees (cum laude) in telecommunication engineering from the Politecnico di Milano, in 2017 and 2019, respectively, where he is currently pursuing the Ph.D. degree in information technology with the Dipartimento di Elettronica, Informazione e Bioingegneria (DEIB). He was a Visiting Researcher with the King's Communications, Learning and Information Processing (KCLIP) Laboratory, King's College London, London, U.K., in 2022. His current research interests include machine learning and localization techniques for vehicular and industrial networks.



STEFANO SAVAZZI (Member, IEEE) received the M.Sc. and Ph.D. degrees (Hons.) in ICT from the Politecnico di Milano, Italy, in 2004 and 2008, respectively. He is currently a Senior Researcher with the Institute of Electronics, Computer and Telecommunication Engineering (IEIIT), Consiglio Nazionale delle Ricerche (CNR). Previously, he was a Visiting Researcher with Uppsala University, in 2005, and the University of California at San Diego, in 2007. He is also a Principal Investigator for the Horizon EU Project Holden. He has coauthored over 100 scientific publications (Scopus). His current research interests include distributed signal processing, federated and distributed machine learning, green networking aspects for the Internet of Things, including radio localization, RF holography, and vision technologies. He has received the Dimitris N. Chorafas Foundation Award, in 2008. He is also serving as an Associate Editor for *Sensors*, *Frontiers in Communications and Networks*, and *Wireless Communications and Mobile Computing*.

MONICA NICOLI (Senior Member, IEEE) received the M.Sc. (Hons.) and Ph.D. degrees in communication engineering from the Politecnico di Milano, Milan, Italy, in 1998 and 2002, respectively. She was a Visiting Researcher with ENI Agip, from 1998 to 1999, and Uppsala University, in 2001. In 2002, she joined the Politecnico di Milano as a Faculty Member. She is currently an Associate Professor in communications engineering with the Department of Management, Economics and Industrial Engineering. She holds the Italian national scientific habilitation as a Full Professor. Her research interests include signal processing, machine learning, and wireless communications, with emphasis on smart mobility, and the Internet of Things (IoT) applications. She was a recipient of the Marisa Bellisario Award, in 1999, and a co-recipient of the Best Paper Award from the IEEE Symposium on Joint Communications and Sensing, in 2021, the IEEE Statistical Signal Processing Workshop, in 2018, and the *IET Intelligent Transport Systems* journal, in 2014. She has served as an Associate Editor for the *EURASIP Journal on Wireless Communications and Networking*, from 2010 to 2017, and a Lead Guest Editor for the Special Issue on Localization of *Mobile Wireless and Sensor Networks*, in 2011. She is also an Associate Editor of the IEEE Transactions on Intelligent Transportation Systems.

Open Access funding provided by 'Politecnico di Milano' within the CRUI CARE Agreement