FedLoRa: IoT Spectrum Sensing Through Fast and Energy-Efficient Federated Learning in LoRa Networks

Fabio Busacca[§], Stefano Mangione[*], Giovanni Neglia[‡], Ilenia Tinnirello[*], Sergio Palazzo[§], Francesco Restuccia[†]
[§] Department of Electrical, Electronics and Computer Engineering (DIEEI), University of Catania, Italy
[*] Department of Engineering, University of Palermo, Italy
[‡] Inria, Université Côte d'Azur, France.
[†] Institute for the Wireless Internet of Things, Northeastern University, Boston, USA

2024 IEEE 21st International Conference on Mobile Ad-Hoc and Smart Systems (MASS) | 979-8-3503-6399-9/24/$31.00 ©2024 IEEE | DOI: 10.1109/MASS62177.2024.00047

***Abstract*—Effective spectrum sensing is quintessential to decrease spectrum congestion across time, space and frequency in Internet of Things (IoT) networks. To circumvent the severe bandwidth constraints of IoT networks, federated machine learning (FML) can be used, but it is still unclear whether FML can be successfully performed in resource-constrained bandwidth-limited IoT networks. In this paper, we demonstrate for the first time that FML can tolerate losses up to a certain percentage and still converge. Then, we leverage this key result to design `FedLoRa`, an optimization networks for LoRa that is (i) fast, as it reduces the FML round time in comparison with other resource allocation schemes; (ii) energy-efficient, as the time reduction does not imply a higher energy consumption. The key idea is to balance the network load over the available spreading factors, and to exploit sequential polling of nodes to maximize the number of simultaneous non-interfering transmissions, leading to a shorter FML round time. As the problem is NP-Hard, we provide an approximation algorithm. We evaluate the performance of `FedLoRa` through experimental evaluation on the Colosseum channel emulator, as well as with real-world data collection with off-the-shelf LoRa devices in an 5kmx5km urban setting in Portland, Maine. Our results show that `FedLoRa` reduces the round time by up to about 35%, as compared to the baselines.**

I. Introduction

The recent growth of the Internet of Things (IoT) – poised to reach 29 billion devices by 2025 – is quickly saturating unlicensed spectrum bands. As spectrum becomes increasingly congested, spectrum sensing operations such as spectrum hole detection (SHD) and radio fingerprinting (RFP) will become crucial to enable secure and effective dynamic spectrum access (DSA) in IoT networks [1]. Prior work has unveiled that deep neural network (DNN) models can be extremely effective in performing SHD and RFP [2]–[4]. At the same time, existing work has shown that the non-stationary, dynamic and unpredictable effect of the wireless channel, as well as hardware-level transceiver impairments, may cause the DNN classification accuracy to plummet [5]. For example, RFP accuracy may decrease by 30% when tested with data collected days after it was trained [6]. To address generalization issues, regularly updating DNN models with data collected in different noise/interference conditions and by multiple devices may significantly increase classification accuracy [6]. While this approach is feasible in high-power wireless networks, the IoT presents a significantly more challenging scenario. Specifically, periodically sending waveform-level I/Q data to a central server far exceeds the capacity of IoT networks. For example, a low-power wide-area network (LPWAN) network such as LoRa has a maximum data rate of 37.5 kbps due to duty cycle and other limitations [7]. By assuming an analog-to-digital converter (ADC) providing I/Q samples with 16-bit resolution, a 100 MHz channel generates 200 MB/s worth of I/Q data, implying that sending a 1-second dataset would take about 11.85 hours.

To address this issue, federated machine learning (FML) has been shown to be very effective in improving the robustness of locally-trained DNNs without the need of sharing datasets to a centralized server. This is ultimately achieved by merging local DNNs into a globally-shared model [8]. Existing work on FML has considered either wired networks and/or high-bandwidth wireless networks [9]–[11]. On the other hand, **enabling fast and efficient FML in the IoT context is a significantly more challenging problem**. Specifically, FML is based on the exchange of local DNN parameters to a central server (also called the *aggregator*). However, as mentioned above, IoT protocols based on the LPWAN paradigm have very limited data rates. Moreover, IoT nodes are severely power-constrained. Thus, on one hand, we want the nodes to send their DNN parameters to the aggregator as fast as possible, to ensure faster FML convergence. On the other hand, we want to keep energy consumption to a minimum to prolong the sensor's lifetime. Although existing literature has investigated energy-aware LoRa optimization [12], [13] and FML techniques for IoT [14], [15], a series of assumptions (e.g., OFDM-based transmissions, zero interference, single-node network) make prior work not entirely applicable to real-world IoT contexts.

This work was partially supported by the European Union under the Italian National Recovery and Resilience Plan (NRRP) of NextGenerationEU, partnership on "Telecommunications of the Future" (PE0000001 - program "RESTART") and by MUR, Italy, under the project PNRR M4 - C2 - investment 1.1: Projects of Significant National Interest (PRIN) - PRIN 2022 code 2022FYCNPT "IoTSensE: IoT-based Sensing Extension"

2155-6814/24/$31.00 ©2024 IEEE
DOI 10.1109/MASS62177.2024.00047

To address the above challenges, this paper makes the following novel contributions:

• We first demonstrate a fundamental theoretical result for FML schemes under lossy wireless communication channels. In particular, we show that, even in presence of random losses of local DNNs parameters at the aggregator, it is possible to derive a *steady-state bound on the global model errors*. Importantly, this bound can be related to the communication error rate experienced by each sensor, and forced to zero in case the learning rate gradually decreases with the number of updating rounds.

• We leverage the theoretical result to design `FedLoRa`, an optimization framework for efficient large-scale federated learning in LoRa wireless networks, in which we optimize the Resource Allocation scheme for transmitting the local models trained by the sensors. We formalize a Resource Allocation Problem for LoRa (LoRa-RAP), where the communication resources (spreading factor and transmission power) assigned to each sensor are optimized so as to minimize the FML round time and keeping energy consumption into account. The key intuition behind this result is to apply a load balancing logic to the resource allocation procedure. In other words, the objective is fairly distribute the network load over the available Spreading Factors (SFs). Usually, nodes sharing the same SF must transmit in a sequential fashion to avoid any possible message collision. Thus, allocating most of the nodes to the same SF results in a long FML round time. Conversely, we spread the nodes over different SFs, so that many simultaneous transmissions can happen at once. This implies a more efficient usage of the network and a shorter FML round time.

• We build a full-fledged prototype of `FedLoRa` and evaluate its performance through extensive experimental evaluation on the Colosseum network emulator [16]. As part of the prototype, we designed and developed (i) a complete Software Defined Radio (SDR) implementation of a LoRa transmitter and receiver; (ii) a generalized GPU-based FML training framework on Colosseum. To the best of our knowledge, we are the first to investigate and evaluate FML-based algorithms in realistic wireless settings;

• We consider a state-of-the-art DNN for SHD [3], and investigate the `FedLoRa` performance with different RF configuration scenarios on Colosseum, as well as with real-world measurements with a LoRa gateway and a LoRa node. Our results show that `FedLoRa` reduces the FML round time by up to about 35% with respect to baselines. **To allow full reproducibility of our results, we pledge to share our code repositories to the community.**

II. Background on LoRa

The Long Range (LoRa) protocol is a low-power wide-area network (LPWAN) modulation technique originally developed by Cycleo and later acquired by Semtech, the founding member of the LoRa Alliance. Specifically, LoRa covers the physical layer (PHY), while LoRaWAN (Long Range Wide Area Network) covers the upper layers (e.g., encryption features) [17]. **Note how the framework presented in this paper leverages on plain LoRa, rather than on the full LoRaWAN stack.** The upper layers are indeed customized and tailored to fit our specific Federated Learning use case.

LoRa networks typically are deployed in a star network configuration, with a centralized LoRa *gateway* receiving data from the LoRa *nodes*. LoRa leverages sub-GHz license-free Industrial, Scientific, and Medical (ISM) bands such as 915 MHz in North America, 868 MHz in Europe, and 923 MHz in Asia, and supports three bandwidth (BW) settings (125kHz, 250kHz, and 500kHz). While wider bandwidths provide higher data rates, they also degrade the receiver sensitivity. LoRa relies on the CSS modulation chirp spread spectrum (CSS), which spreads the signal over a broader frequency band to attain interference resiliency in ISM bands and increase the communication range. Specifically, a LoRa symbol is one chirp spanning the entire bandwidth. By defining SF as the *spreading factor*, the chirp is modulated by cyclically shifting its instantaneous frequency by one of 2^{SF} values to carry SF binary symbols. In particualr, LoRa supports up to six different SFs ranging from 7 to 12. Increasing the SF extends the coverage at the expense of reducing the data rate as the packet duration will also increase. For example, if SF equals 7 (each symbol carries seven bits but symbol lasts 2^7 chips), then the length of the chirp is $2^{\mathrm{SF}} = 128$ chips and the data rate $7/128$ bits per chip = 6.8 kbps. Techniques based on forward error correction (FEC) can further improve the receiver sensitivity. LoRa supports four channel coding rate values between 0.5 and 0.8, computed by $4/(4+\mathrm{CR}), \mathrm{CR} \in \{1,2,3,4\}$. A bigger CR enhances the protection but lowers the information bit rate.

III. The Federated Learning Scheme

We defined a federated learning scheme suitable for LoRa networks, in which communication links have limited bandwidth and are error prone. We assume that N AI-ready IoT devices (e.g., Nvidia Jetson series embedded computing boards), also called clients, work each on a local data-set to be used for training a model characterized by a vector of d parameters $\boldsymbol{w} \in \mathbb{W} \subset \mathbb{R}^d$, where $\mathbb{W}$ denotes the hypothesis class. Let $S_i = \{x_{i,l}\}_{l=1}^{|S_i|}$ denote the data-set available to the i-th client with size $|S_i|$, and $S = \cup_{i=1}^{N} S_i$ denote the total data-set. Let $f(\boldsymbol{w}, x)$ denote the loss of model $\boldsymbol{w}$ on sample x and $f(\boldsymbol{w}, B) = 1/|B| \sum_{x \in B} f(\boldsymbol{w}, x)$ denote the average loss on the set B. For simplicity we also use $f_i(\boldsymbol{w})$ to indicate the average loss on the client-i's dataset, i.e., $f_i(\boldsymbol{w}) = f(\boldsymbol{w}, S_i)$. The goal of the training process is to find a minimizer $\boldsymbol{w}^* \in \mathbb{W}$ of an opportune weighed sum of clients' local losses:

$$\boldsymbol{w}^* = \arg\min_{\boldsymbol{w} \in \mathbb{W}} \sum_{i=1}^{N} \alpha_i f_i(\boldsymbol{w}) \triangleq F(\boldsymbol{w}) \tag{1}$$

where $\{\alpha_i\}_{i=1}^{N}$ are weighing parameters, and can be properly set, for instance, to ensure per-client fairness ($\alpha_i = \frac{1}{N}$) or per-sample fairness ($\alpha_i = \frac{|S_i|}{|S|}$).

Our algorithm is an adaptation of the FedAvg algorithm [18]. The process is organized in a set of communication

rounds, in which each client locally executes multiple update epochs before sending back the local weights to the central server. More into details, at a generic round k the process works as described below.

Broadcasting phase: the central server transmits in downlink the current model parameter vector $\boldsymbol{w}^{k,0}$ to all clients.

Local update: each client i performs multiple gradient updates of the local model over a set of E random batches of samples $B_i^k = \{B_i^{k,e}\}_{e=1}^E$ drawn from the local data-set. Given the per-round learning rate η^k, each client computes

$$\boldsymbol{w}_i^{k,e+1} = \boldsymbol{w}_i^{k,e} - \eta^k \nabla f_i(\boldsymbol{w}_i^{k,e}, B_i^{k,e})$$

for $e = 0, 1, \ldots, E-1$. The final model update, $\boldsymbol{w}_i^{k,E}$, is sent back to the central server by transmitting, in general, multiple LoRa frames. The frames are organized by including randomly ordered components, together with the relevant model index, in order to allow an exact identification of the updated parameters at the server. Therefore, we assume that losses due the corruption of one packet are uniformly spread on the total number of model dimensions d.

Central update: At the end of each communication round, the central server aggregates all the parameters received by the clients. Let $\hat{\boldsymbol{w}}_i^{k,E}$ be the local models available at the aggregator, which are generally different from $\boldsymbol{w}_i^{k,E}$ because of transmission errors. We assume to replace any missing component $[\boldsymbol{w}_i^{k,E}]_l$ sent by the i-th client with the component $[\boldsymbol{w}_i^{k-1,0}]_l$ from the previous step. This replacement occurs with probability CER_i, CER_i being the *Component Error Rate* associated to client i. In order to compensate for the learning bias introduced by the component loss, assuming that the server has an estimate of the CER_i experienced by client i, the global model is updated as:

$$\boldsymbol{w}^{k+1,0} = \Pi_{\mathbb{W}}\left(\boldsymbol{w}^{k,0} + \sum_{i=1}^{N} \frac{\alpha_i}{1-CER_i}\left(\hat{\boldsymbol{w}}_i^{k,E} - \boldsymbol{w}^{k,0}\right)\right) \quad (2)$$

where $\Pi_{\mathbb{W}}$ denotes the Euclidean projection on $\mathbb{W}$, i.e., $\Pi_{\mathbb{W}}(\boldsymbol{w}') = \arg\min_{\boldsymbol{w}\in\mathbb{W}} \|\boldsymbol{w} - \boldsymbol{w}'\|$. Instead of simply averaging the current local models as in FedAvg, in (2) the global model is updated by a pseudo-gradient as in [19], [20].

A. Convergence

In this section we investigate the convergence of our FML scheme in case of losses on the communication links.

Note how, in our scheme, there are two main sources of randomness: i) random batch selection at each node, and ii) channel losses. We make the following assumptions:

Assumption 1: $\mathbb{W}$, the hypothesis class (i.e., the set of possible parameter vectors), is convex and compact with diameter D, and contains $\boldsymbol{w}^*$ in its interior.

Assumption 2 (L-Smoothness): *The local loss functions* $\{f_i\}_{i=1}^N$ *have L-Lipschitz continuous gradients:* $\|\nabla f_i(\boldsymbol{w}) - \nabla f_i(\boldsymbol{w}')\| \leq L\|\boldsymbol{w} - \boldsymbol{w}'\|, \ \forall \ \boldsymbol{w}, \boldsymbol{w}' \in \mathbb{W}$.

Assumption 3 (Convexity): *The local loss functions* $\{f_i\}_{i=1}^N$ *are convex:* $f_i\Big(\alpha\boldsymbol{w} + (1-\alpha)\boldsymbol{w}'\Big) \leq \alpha f_i(\boldsymbol{w}) + (1-\alpha) f_i(\boldsymbol{w}'), \forall \ \boldsymbol{w}, \boldsymbol{w}' \in \mathbb{W}, \forall \ \alpha \in [0,1]$.

Assumption 4 (Bounded variance): *The variance of the local estimated gradients* $\nabla f_i(\boldsymbol{w}_i^k, B_i^k)$ *due to the random sampled data is bounded by* σ^2: $\mathbb{E}_{B_i^k}\Big[\|\nabla f_i(\boldsymbol{w}_i^k, B_i^k) - \nabla f_i(\boldsymbol{w}_i^k)\|^2\Big] \leq \sigma^2$.

Assumption 5 (Losses): *Transmission losses occur only upstream and are independent across clients and over time*.

From Assumptions 1, 2, and 4 it follows that $\exists G^2 (= \sigma^2 + L^2 D^2)$ such that $\mathbb{E}_{B_i^{k,e}}\Big[\|\nabla f_i(\boldsymbol{w}_i^k, B_i^{k,e})\|^2\Big] \leq G^2$. Let f_i^* be the global minimum of f_i. Similarly to other works [19], [21], $\Gamma = \sum_{i=1}^N \alpha_i f_i^* - F(\boldsymbol{w}^*)$ quantifies the heterogeneity of clients' local data-sets.

Theorem 1. *If we select* $\eta^k = \frac{\eta}{k^\alpha}$ *with* $\alpha \in (\frac{1}{2}, 1)$*, and* $\eta \leq \frac{1}{2L(2E+1)}$*, then:*

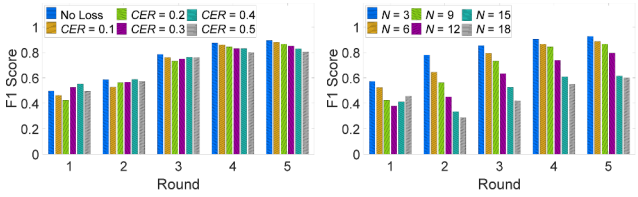
$$\mathbb{E}\Big[F(\overline{\boldsymbol{w}}^{K,0}) - F(\boldsymbol{w}^*)\Big] \leq \frac{2}{E\sum_{k=1}^K \eta^k}\Bigg\{\|\boldsymbol{w}^{1,0} - \boldsymbol{w}^*\|^2 + C\Big[E^2G^2\sum_{i=1}^N \alpha_i \frac{CER_i}{1-CER_i} + 4LE(1+LE)\Gamma + \frac{3}{2}\sigma^2E^2 + \frac{1}{6}E(E-1)(2E-1)G^2\Big]\Bigg\},$$

where $\overline{\boldsymbol{w}}^{k,0} = \frac{\sum_{k=1}^K \eta^k \boldsymbol{w}^{k,0}}{\sum_{k=1}^K \eta^k}$ *and* $\sum_{k=1}^K (\eta^k)^2 \leq C \ (< +\infty$*, for any* K*).*

The proof for Theorem 1 is in the Appendix A. As $\sum_{k=1}^K \eta^k \in \Theta\left(K^{1-\alpha}\right)$, the theorem shows that the optimality gap $F(\overline{\boldsymbol{w}}^{K,0}) - F(\boldsymbol{w}^*)$ converges to 0 in expectation. In particular, the convergence rate can be made arbitrarily close to $K^{-1/2}$ by choosing smaller values of α. We remark how higher data-set heterogeneity (larger Γ) and higher levels of noise due to batch sampling (larger σ^2 and G^2) and to trasmission losses (larger $\{CER_i\}_{i=1}^N$) negatively affect convergence. In particular, for $CER_i \to 1$, the bound diverges, corresponding to the fact that client i does not succeed in participating in the training, i.e., no information about f_i is available to the server. All in all, Theorem 1 describes the relationship between the Federated Learning model and the system parameters, including the CER.

Figure 1 shows a numerical example of our federated learning scheme. The reference machine learning (ML) model for our experiments is the neural network from [3], whose specific aim is the detection of Spectrum Holes in LTE cellular networks. The overall size of the network is 62 kBytes, and can thus suit the low data-rate capabilities of LoRa. The total number of weights in the network is 15,888. Since each weight is represented as a 4 byte float, a single LoRa packet can fit up to 62 weights. In our experiments, we assume that: i) the dataset is equally split among all clients, with an overlapping percentage of 10%, which makes the data distribution strongly non-i.i.d; ii) all clients experience the same CER value. Figure 1a depicts the F1 score of the global model as the CER varies in the range $\{0, 0.1, 0.2, 0.3, 0.4, 0.5\}$ Such an error rate varies as a function of the interference level experienced in

the network (i.e., on the number and position of nodes which can transmit simultaneously). Specifically, we observe that the final F1 score of the model after 5 rounds decreases as the CER gets higher. Notably, when the $CER = 0.1$, the final score is only minimally impacted by the CER. Accordingly, in Section 4.1 we decide to target a CER equal at most to 10% for all clients, as a trade-off between the acceptable interference level and the convergence speed of the training process. Figure 1b shows how the F1 score is affected by the number of involved nodes, and, consequently, by the level of fragmentation of the dataset, for a fixed $CER = 0.3$. The more fragmented is the dataset, the less is the F1 score after five rounds. The impact of the fragmentation becomes critical for the case for $N = 15$ and $N = 18$, causing a significant drop in the F1 score. In fact, the more are the nodes and the dataset fragmentation, the more rounds are required to converge to an acceptable model performance. An interesting insight is that the more fragmented is the dataset, the lower should be the target CER, to compensate the performance drop induced by the fragmentation.



(a) Variable CER, 9 nodes (b) Variable number of nodes, CER = 0.3

Fig. 1: F1 score of the DeepSense model [3] with our proposed FML scheme.

IV. The FedLoRa Framework

The main target of FedLoRa is to establish reliable, effective and energy-efficient federated learning in LoRa networks, taking into account the channel and interference conditions of the sensors (i.e. the network level) and the acceptable errors on the global model over time (i.e. the learning level). We now provide a walkthrough of FedLoRa below with the help of Figure 2. At the beginning, the architecture of the DNN model $M = \boldsymbol{w}$ is shared with FedLoRa (**Step 1**). The DNN model weights are then forwarded to the LoRa *nodes* by the LoRa *gateway* (**Step 2**). Then, the DNN model size is fed to the LoRa Resource Allocation Problem (LoRa-RAP), formulated in Section IV-A. The LoRa-RAP takes as input some channel-related information, such as signal-to-noise ratio (SNR) and signal-to-interference (SIR) ratio, which are estimated experimentally through pilot transmissions (**Step 3**), and the desired CER on the model components. The PHY parameters are then sent to each node in the network through the LoRa *gateway* (**Step 4**). As regards the FML training, each node trains a local DNN model in several subsequent *rounds*. Since nodes are not computationally powerful, the local model for node i, $M_i = \boldsymbol{w} - \eta \nabla f_i(\boldsymbol{w})$, is trained only with a random sampled batch B_i of the locally-available data. Each node then transfers the weights of M_i to the gateway. The weights received from all the nodes are aggregated for updating the global model, which is then sent back to the nodes for the next round (**Step 5**).

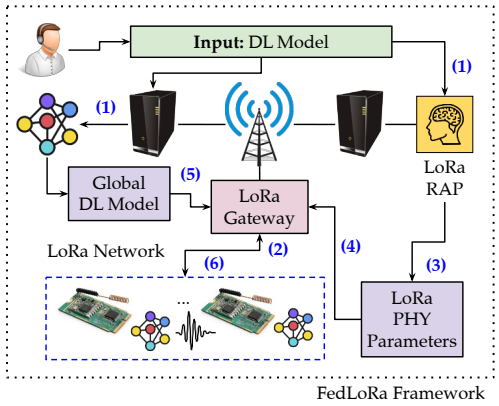


Fig. 2: High-level overview of the FedLoRa framework.

A. LoRa-RAP Formulation

We assume that the network configuration can be optimized by dynamically adjusting the transmission power and SF used by each sensor. Higher SFs are associated to more robust links, yet to lower data-rates (and, thus, to a longer transmission time). Moreover, higher values of transmission power guarantee better SNR at the receiver, but also imply a bigger energy consumption, and produce more interference to other nodes in the network. Other communication parameters such as the transmission bandwidth B and coding rate CR are not considered as network tunable parameters. In real LoRa applications the bandwidth is not configurable. For example, according to EU LoRa regulations, all SFs can only work on a bandwidth of 125 kHz (except for SF7). Moreover, we do not include the CR PHY parameter in the optimization as we assume a Line-of-Sight channel model (i.e. Rice fading), and a very limited effect of the coding gain employed in LoRa.

Since each sensor has to transmit a bulk of frames at each model update round, which results in a temporary congestion of the network[1], as a first optimization strategy we consider the possibility of polling the network nodes sequentially rather than using random access. Multiple polling rounds are executed in parallel on different SFs, by exploiting the capability of LoRa modulation of rejecting signals modulated at different SFs even for negative SIR values. By considering this access solution, we set the signal to noise (SNR) and interference (SIR) ratios which need to be provided to each node for guaranteeing the desired CER. For a reference node n, we assume that interfering signals are only due to other LoRa nodes transmitting at a SF different from SF_n (which are not perfectly orthogonal [22]) and that the channel is AWGN.

The goal of LoRa-RAP is to find the values of SF and transmission power for each node able to minimize the FML

[1]In this work, we assume to work on a channel without duty-cycle limitations (869.7MHz).

round time, while also taking into account the aforementioned constraints. We define $\mathcal{N}$ as the set of LoRa nodes in our reference scenario and $\mathbf{SF}$ as the set of available spreading factors. Moreover, ρ_n and SF_n are the transmission power and the spreading factor for a generic node n, respectively. We now model the transmission time in the system. First, as in [23], the data-rate of a LoRa node n is:

$$r_n = \mathrm{SF}_n \frac{B}{2^{\mathrm{SF}_n}} \frac{4}{4+\mathrm{CR}}$$

Accordingly, being s the size of the model weights (in bytes) to be transmitted, the transmission time of node n is equal to:

$$T_{SF_n,n} = \frac{s}{r_n} = s \frac{2^{\mathrm{SF}_n}}{B \cdot \mathrm{SF}_n} \frac{4+\mathrm{CR}}{4}$$

where we assume that the model size s is fixed for all the nodes[2]. We define the resource allocation problem as follows.

LoRa Resource Allocation Problem (LoRa-RAP)

$$\min_{\boldsymbol{\rho},\mathbf{SF}} \left\{ \max_{\mathrm{SF},x} |N_{\mathrm{SF},x}| T_{\mathrm{SF},x} \right\} \quad (3)$$

s.t.

$$\rho_{min} \leq \rho_n \leq \rho_{max} \ \forall n \in \mathcal{N} \quad (4)$$

$$\mathrm{SF}_n \in \{7,8,9,10,11,12\} \ \forall n \in \mathcal{N} \quad (5)$$

$$\mathrm{SIR}_n(\mathrm{SF}_n, \mathrm{SF}_{int}) \geq \mathrm{SIR}_{th}(\mathrm{SF}_n, \mathrm{SF}_{int}) \ \forall n \in \mathcal{N}, \\ \forall \ \mathrm{SF}_{int} \in \{7,8,9,10,11,12\}, \mathrm{SF}_{int} \neq \mathrm{SF}_n \quad (6)$$

$$\mathrm{SNR}_n \geq \mathrm{SNR}_{th}(\mathrm{SF}_n) \ \forall n \in \mathcal{N} \quad (7)$$

where:

- $N_{\mathrm{SF},x}$ is the number of users assigned to SF x;
- $\mathrm{SIR}_n(\mathrm{SF}_n, \mathrm{SF}_{int})$ is the worst-case Signal to Interference Ratio of node n w.r.t. any interfering transmission performed at spreading factor SF_{int} at the gateway. In other words, it measures the interference produced by the strongest interfering node on SF_{int};
- SNR_n is the signal to noise ratio of node n at the gateway.

With reference to constraint (6), whenever SIR_n is below the threshold $\mathrm{SIR}_{min}(\mathrm{SF}_n, \mathrm{SF}_{int})$, decoding of weights sent from node n may fail with a given CER that depends on the chosen threshold. Hence, we properly tune the threshold values to guarantee the desired CER of 10% on the incoming packets transmitted at each SF. The reference values are shown in Table I. The reference threshold values are derived as in [22].

For what concerns constraint (7), the threshold value $SNR_{th}(SF_n)$ corresponds to the minimum SNR required to correctly decode the incoming frames. The SNR threshold values are set as in [24].

[2]A further direction for optimizing the system could be the usage of compression schemes for sending the model weights, which could lead to heterogeneous model sizes s_n.

The problem above formulated is non-linear and includes continuous variables ($\boldsymbol{\rho}$), integer variables ($\mathbf{SF}$), as well as non-linear constraints (constraints 6 and 7), and therefore falls into the category of Mixed-Integer Nonlinear Programming (MINLP) problems. More in detail, LoRa-RAP can be assimilated to a Parallel Machine Scheduling (PMS) problem [25]. A PMS problem involves the minimization of the maximum completion time, i.e., the *makespan*, for a given set of n tasks. The tasks should be properly allocated among m parallel computing machines according to several constraints, e.g., the allowed maximum processing time and the required minimum processing power. In such a perspective, the LoRa nodes are the machines of the PMS problem, the set of SFs corresponds to a set of tasks, the round time is the makespan, and the sensitivity and interference constraints correspond to the processing power requirements. Since PMS problems are provably **NP-Hard**, LoRa-RAP falls into this category of problems, and accordingly can be solved through either numerical approximation or greedy algorithms.

B. A Greedy Algorithm for LoRa-RAP

We designed a greedy algorithm for resource allocation. In particular, the algorithm is based on a key-intuition: the round time can be minimized if the network nodes are properly distributed among the available SFs. Hence, we introduce the following load-balancing constraint, which aims at balancing the load on all the available SFs:

$$N_{\mathrm{SF},x} = N \frac{\left(1/AT_{\mathrm{SF},x}\right)}{\sum_{\mathrm{SF}\in\mathbf{SF}} \left(1/AT_{\mathrm{SF}}\right)} \forall x \in \mathbf{SF} \quad (8)$$

Hence, the smaller is the air time, the bigger is the portion of users assigned to the corresponding SF. The main reason to perform load balancing is indeed to maximize the parallel (non-interfering) transmissions and, consequently, to reduce the FML round time.

The algorithm is split in several phases: (i) Estimate the channel, (ii) Meet SNR constraints, (iii) Meet SIR constraints. During the first phase, the gateway node estimates the maximum SNR of each node (we assume the channel gain matrix and signal noise power to be known a priori). The maximum SNR can be easily estimated by assuming the nodes to transmit at the maximum available power. Nodes are accordingly ordered by SNR, in descending order. In the second phase, each node is assigned to the minimum SF possible, in compliance with the SNR constraint (7). However, if this choice violates the load-balancing constraint (8), the algorithm tries to assign

SF \ SF_{int}	7	8	9	10	11	12
7	0.1	-7.0	-8.7	-9.6	-10.2	-11.0
8	-10.4	-0.3	-10.2	-12.1	-12.8	-13.2
9	-14.2	-13.3	-0.6	-13.0	-14.4	-15.7
10	-16.9	-17.2	-15.9	-0.9	-16.1	-17.9
11	-19.7	-19.8	-20.0	-19.0	-1.3	-19.2
12	-22.2	-22.6	-22.8	-23.2	-22.1	-1.9

TABLE I: SIR thresholds for a 10% Packet Error Rate, maximum size payload, $\mathrm{CR} = 1$

the node to the next higher SF. This procedure is repeated until either constraint (8) is satisfied, or until the maximum SF value is reached. The last phase focuses on the SIR constraint (6).

For each node n in the scenario, the algorithm checks if the assigned transmission power is compliant with the SIR constraint for SF_n. If not, the algorithm proceeds to increase the transmission power, until either the constraint is satisfied, or the assigned power exceeds the maximum power limit. In the latter case, no feasible solution can be found. Then, the algorithm starts over and repeats the check – since the nodes are examined in a sequential fashion, adjusting the transmission power of node i could lead to a violation of the SIR constraint for at least one of the $i-1$ previous nodes. Hence, the algorithm needs to repeat the check and exits if and only if the SIR constraint is still satisfied for all nodes. If not, the algorithm executes step 3 several times. If the algorithm converges, it yields a sub-optimal solution to the round time minimization problem formulated in (3).

C. `FedLoRa` Prototype

We prototyped and evaluated `FedLoRa` and LoRa-RAP on Colosseum, the world's largest network emulator [16]. Colosseum is a wireless emulator with 128 Standard Radio Nodes (SRNs). Each SRN is equipped with 48-core Intel Xeon E5-2650 CPUs and an NVIDIA Tesla K40m GPU, and with a NI/Ettus USRP X310 SDR as well. The SRNs are all linked together by a Massive Channel Emulator (MCHEM), which is responsible for the emulation of the wireless channels. Thanks to its FPGA modules, the MCHEM processes the radio signals through Finite Impulse Response (FIR) filters, and thus emulate the effects of a real radio channel, such as attenuation and propagation delay.

As part of the prototype, we implemented from scratch the first full-fledged LoRa PHY layer for SDR platforms, able to guarantee compatibility with commercial LoRa-based sensors. To this purpose, we performed a low-level analysis of real signals, transmitted by LoRa commercial devices, for solving a few modulation ambiguities of the LoRa patent from Semtech. The full implementation of our SDR LoRa transceiver is presented in [26], and is available on GitHub[3].

Besides the LoRa PHY, we implemented for the first time a MAC protocol to establish reliable data exchange between the nodes and the gateway based on polling. Indeed, for transmitting the gradient components of the local model at each communication round, sensors generate a bulk of data frames. Our polling mechanism is intended to replace the ALOHA mechanism of standard LoRa networks, which has a very limited efficiency in case of greedy traffic sources. With a polling mechanism, nodes assigned to the same SF transmit in a sequential way, and do not interfere with each other, while nodes working on different SFs can be polled in parallel. **For this reason, balancing the load in the network is of crucial importance** for maximizing parallel (non-interfering) transmissions, thus reducing the FML round time.

[3]https://github.com/fabio-busacca/sdr-lora

V. Colosseum Results

Before presenting the results obtained through Colosseum, we first describe the baselines and the RF scenarios.

Baselines. We compare LoRa-RAP with:

- *MinPower*: this algorithm first focuses on the minimization of the node transmission power. Then, it assigns the lowest SF possible, in compliance with LoRa SNR requirements.
- *BestSF*: as opposed to *MinPower*, the main goal of *BestSF* is to first find the minimum SF allowed by the SNR constraints, and, only then, to minimize the transmission power. No load balancing criteria are applied.

Both algorithms also deal with the SIR constraints in (6). The adopted procedure is identical to the one of LoRa-RAP.

Scenario Description. We now describe the custom LoRa scenarios implemented in Colosseum to evaluate LoRa-RAP. A network scenario is easily defined in Colosseum as a collection of wireless links between several radio nodes. Each link is specified by digital channel taps, which are fed to the MCHEM at run time. As depicted in Figure 3, the scenario involves a maximum of 18 LoRa nodes and one LoRa gateway, scattered over a $400\times400m^2$ area. The color scheme is related to the SF value assigned to each node by LoRa-RAP in the full setting (as the number of considered nodes varies, the allocation of resources is accordingly different).

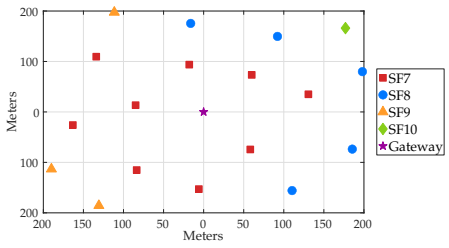


Fig. 3: Colosseum Testbed: Location of the LoRa Nodes and of the Gateway in the emulated scenario. Each node is associated to a SF value through the LoRa-RAP algorithm

Channel attenuation is calculated by means of Friis propagation model, with a path loss exponent $\alpha = 2$ (i.e. the coefficient for free space path loss scenarios). Note that the choice to simulate a "small" area and to assume a free space propagation model could seem unsuitable for the emulation of LoRa-based communications. This choice, however, is due to the intrinsic limitations of Colosseum, and, more specifically, of the SDR hardware, which introduces a noise power estimated as high as $\sigma_n^2 \approx 3,5 * 10^{-8}$. This effect naturally reduces the maximum allowed simulated attenuation and, as a consequence, the maximum simulated communication distance.

Figure 3 depicts the virtual locations of the nodes in the emulated scenario, as well as the SF resource allocation performed by LoRa-RAP. Note how the farthest nodes are naturally associated to higher SFs. Moreover, the SF allocation reflects the load balancing criterion previously described.

Experiments. The experiments performed on the Colosseum channel emulator aimed at the evaluation of two main metrics: **FML round time**, and **energy consumption**. Note how the latter represents a *normalized* energy value, rather than a real energy value. Indeed, since SDRs are uncalibrated devices, calculating the actual output power is not possible. Instead, we calculate the energy consumption for each node from the digital signal amplitude, and from the model transmission time. As already stated in Section III, the reference ML model for our experiments is the neural network from [3]. The experiments were run for several sub-sets of active nodes, ranging from just three LoRa nodes, to the full configuration of eighteen nodes. In each configuration, the chosen nodes were the closest to the gateway. Finally, the FML round time and the energy consumption have been averaged on a total of five federated rounds per experiment.

Note how, in this particular scenario, the baseline *MinPower* allocates all the nodes to SF 9, while, for *BestSF*, every node is assigned to SF 7.

Figure 4 depicts the average FML round time for LoRa-RAP and the baselines. The results show how our approach is able to reduce the average time for a FML round, and, hence, to finally reduce the convergence time of the FML procedure. In fact, while *BestSF* simply assigns the smallest SF possible to each node, LoRa-RAP balances the load among the available SFs, allowing for simultaneous transmissions on different SFs, thus reducing the FML round time by up to about 35%. The baseline *MinPower* offers instead the worst performance: since the priority is the minimization of the transmission power, the nodes are allocated to higher SFs (in this specific case, SF 9), resulting in a high transmission time.

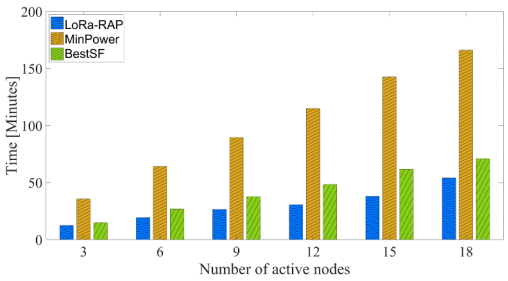


Fig. 4: Colosseum-LoRa-RAP vs baselines: FML Round Time

Figure 5 reports the average per-node energy consumption for LoRa-RAP and the baselines. Significantly, all three strategies exhibit a similar performance for each node configuration, with a slight reduction in the energy consumption for the *BestSF* strategy. Note how a bigger number of active nodes results in a bigger average energy consumption. Indeed, the farthest nodes either transmit on higher SFs and/or with high transmission power. Hence, the average energy consumption accordingly increases.

In conclusion, Figures 4 and 5 highlight how LoRa-RAP is able to sensibly reduce the FML round time without impacting on the average per-node energy consumption.

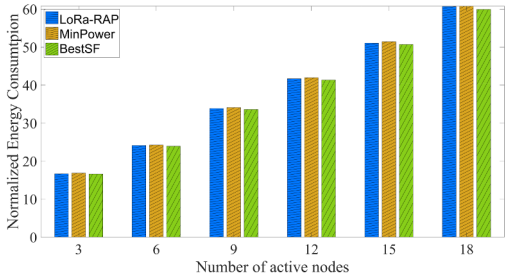


Fig. 5: Colosseum - LoRa-RAP vs baselines: Average per-node Energy Consumption

VI. Testbed Results

To validate the performance of `FedLoRa` with real-world data on a larger scale, we have collected realistic SNR and RSSI values from real-world measurements through the testbed depicted in Figure 6. Specifically, we used (i) one Adafruit RFM95W LoRa radio transceiver breakout board operating at 915MHz, equipped with a Semtech SX1276 Engine with 127 dB Dynamic Range RSSI; (ii) a LoRa-compliant RAK7268C WisGate Edge Lite 2 gateway from RAK Wireless; (iii) an NVIDIA Jetson Nano. We placed the LoRa gateway inside our laboratory and collected several SNR/RSSI measurements in several different locations, at a maximum distance of about 5 Km from the gateway, as shown in Figure 6. The SNR and RSSI values have been used to model the channel conditions between the nodes and the gateway, with an estimated noise power of about -105 *dbm*. All these data has been fed to LoRa-RAP and to the baseline allocation procedures. Since results from the previous section demonstrated the ineffectiveness of *MinPow*, we omit the performance results from this baseline. Instead, we introduce a variant of LoRa-RAP called LoRa-RAP Min SF. The difference is in the way the load balancing criterion is applied. While LoRa-RAP tries to distribute the nodes over all the available SFs (if possible), LoRa-RAP Min SF also aims at keeping the SFs as low as possible. Intuitively, in small-scale scenarios, LoRa-RAP Min SF performs similarly to vanilla LoRa-RAP. For this reason, we chose not to evaluate LoRa-RAP Min SF on Colosseum, and, instead, to test its performance over a larger, simulated scenario. Once again, the baselines are evaluated in terms of both average FML round time, and energy consumption. Similarly to the experiments run on Colosseum, we evaluate the allocation strategies over a variable number of nodes, ranging from 24 to 42 nodes. Figure 6 illustrates the SF allocation for LoRa-RAP over all the 42 evaluated positions.

Figure 7 depicts the FML round time. LoRa RAP and LoRa-RAP Min SF achieve a relevant improvement in the FML round time. More specifically, LoRa-RAP is the best allocation strategy in most configurations, with an improvement of up to 50% over *BestSF*. However, in some specific cases, LoRa RAP Min SF introduces a slight improvement in the FML round time, as compared to its vanilla version. The reason is the following: in some specific cases, the node number can

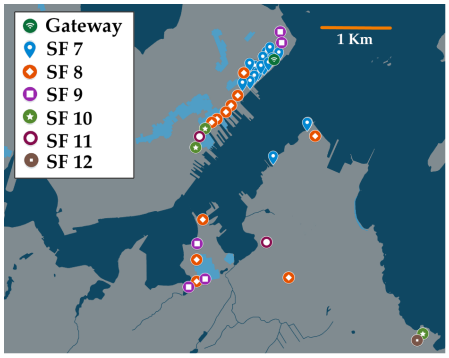


Fig. 6: Locations of the nodes during the data collection.

not exactly match the load balancing criteria. For instance, let us consider a network made up of 8 nodes. if the airtime over SF x is always double the airtime over SF $x+1$, i.e. $AT_{\mathrm{SF},x+1} = 2 * AT_{\mathrm{SF},x}$, then the load balancing condition is $\{4x, 2x, x\}$, i.e. 4 nodes assigned to SF 7, two nodes to SF 8, and one to SF 9. In this case, however, this condition can not be satisfied, and the resulting allocation is instead $\{4, 2, 2\}$. The maximum transmission time, therefore, is $2 * AT_{SF,9}$. Under proper channel conditions, one extra node could instead be allocated to SF 8, resulting in the distribution $\{4, 3, 1\}$, and in a smaller transmission time of $3 * AT_{\mathrm{SF},8} = 1.5 * AT_{\mathrm{SF},9}$.

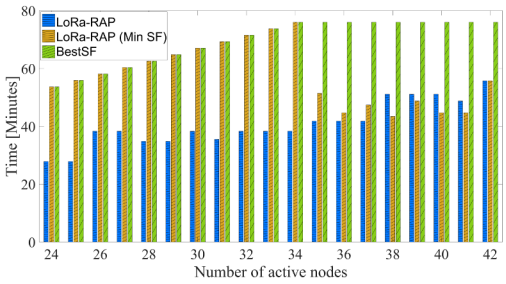


Fig. 7: Testbed - LoRa-RAP vs baselines: FML Round Time

Figure 8 reports the average energy consumption per-node. In line with the results from Colosseum, LoRa-RAP and LoRa-RAP Min SF exhibit a slightly higher energy consumption, as compared to *BestSF*. Once again, including more nodes, and, specifically, the farthest ones, results in an increased average energy consumption.

The results show how LoRa-RAP is able to sensibly reduce the FML round time with little to no impact on the average per-node energy consumption.

VII. Conclusions

We have presented `FedLoRa`, an optimization framework for fast and efficient Federated Learning in IoT LoRa wireless networks. We have first demonstrated a fundamental theoretical result for federated learning schemes with communication errors, and demonstrated that the related learning error can be forced to zero in case the learning rate gradually decreases with the number of updating rounds. Then, we leveraged this important result in the formalization of the LoRa Resource Allocation Problem (LoRa-RAP). In LoRa-RAP, the communication resources (spreading factor and transmission power) assigned to each LoRa sensor are optimized according to a load balancing logic to achieve an improved FML round time while reducing energy consumption. Since the proposed optimization problem is NP-Hard, we have provided an approximation algorithm to solve it in polynomial time. We have built a full-fledged prototype of `FedLoRa` and evaluated its performance through extensive experimental evaluation on the Colosseum emulator [16]. To the best of our knowledge, we are the first to investigate and evaluate FML-based algorithms in realistic IoT settings. We have considered one state-of-the-art DNN model for SHD [3], and investigated the `FedLoRa` performance on Colosseum, as well as with real-world measurements with a LoRa gateway and a LoRa node. Our results have shown that `FedLoRa` reduces the FML round time by up to about 35% with respect to baselines. To allow full reproducibility of our results, we pledge to share our code to the community.

Appendix

We introduce the following notation:

$$\boldsymbol{g}^k = \sum_{i=1}^{N} \sum_{e=0}^{E-1} \alpha_i \nabla f_i(\boldsymbol{w}_i^{k,e}, B_i^{k,e}), \tag{9}$$

$$\overline{\boldsymbol{g}}^k = \sum_{i=1}^{N} \sum_{e=0}^{E-1} \alpha_i \nabla f_i(\boldsymbol{w}_i^{k,e}), \tag{10}$$

$$\boldsymbol{w}_{\mathrm{id}}^{k+1} = \boldsymbol{w}^{k,0} + \sum_{i=1}^{N} \alpha_i (\boldsymbol{w}_i^{k,E} - \boldsymbol{w}_i^{k,0}). \tag{11}$$

Note that, ignoring the projection, $\boldsymbol{w}_{\mathrm{id}}^{k+1}$ would coincide with the global model under ideal lossless transmissions ($CER_i = 0$ for all i). Let $B^k = \{B_1^k, B_2^k, \cdots B_N^k\}$ denote the batches drawn during communication round k, and T^k denote the channel losses occurred during round k. In what follows, when we write an expectation with respect to variables at round $k+1$, we implicitly condition on the previous history, i.e., on $H^k = \{B^1, T^1, B^2, T^2, \ldots, B^k, T^k\}$.

The proofs of the first three Lemmas can be easily obtained adapting the proofs of Lemma 1, 2, and 3 in [21].

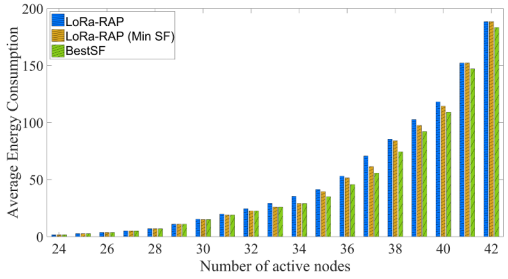


Fig. 8: Testbed - LoRa-RAP vs baselines: Average per-node Energy Consumption

Lemma 1. *Under Assumptions 2–4, if* $\eta_k \leq \frac{1}{2L(2E+1)}$*, then:* $\mathbb{E}_{B^k}\Big[\|\boldsymbol{w}_{id}^{k+1,0} - \boldsymbol{w}^*\|^2\Big] \leq \|\boldsymbol{w}^{k,0} - \boldsymbol{w}^*\|^2 - \frac{1}{2}\eta^k E\sum_{i=1}^N \alpha_i\Big(f_i(\boldsymbol{w}^{k,0}) - f_i(\boldsymbol{w}^*)\Big) + \mathbb{E}_{B^k}\Big[\sum_{i=1}^N \alpha_i \sum_{e=0}^{E-1}\|\boldsymbol{w}_i^{k,e} - \boldsymbol{w}^{k,0}\|^2\Big] + 4(\eta^k)^2\Big(LE(1+LE)\Gamma + \sigma^2E^2 + \mathbb{E}_{B^k}\Big[\|\overline{\boldsymbol{g}}^k - \boldsymbol{g}^k\|^2\Big]\Big)$.

Lemma 2. *Under Assumption 4,* $\mathbb{E}_{B^k}\Big[\|\overline{\boldsymbol{g}}^k - \boldsymbol{g}^k\|^2\Big] \leq \frac{1}{2}E^2\sigma^2$.

Lemma 3. *Under Assumption 4,* $\mathbb{E}_{B^k}\Big[\|\sum_{i=1}^N \alpha_i \sum_{e=0}^{E-1}\|\boldsymbol{w}_i^{k,e} - \boldsymbol{w}_i^{k,0}\|^2\Big] \leq \frac{1}{6}(\eta^k)^2E(E-1)(2E-1)G^2$.

Lemma 4. *Under Assumptions 1, 4 and 5,* $\mathbb{E}_{B^k,T^k}\Big[\|\boldsymbol{w}^{k+1,0} - \boldsymbol{w}^*\|^2\Big] \leq (\eta^k)^2E^2G^2\sum_{i=1}^N \alpha_i\frac{CER_i}{1-CER_i} + \mathbb{E}_{B^k}\|\boldsymbol{w}_{id}^{k+1,0} - \boldsymbol{w}^*\|^2$.

Proof. $\|\boldsymbol{w}^{k+1,0} - \boldsymbol{w}^*\|^2 \leq \|\boldsymbol{w}^{k,0} + \sum_{i=1}^N \frac{\alpha_i}{1-CER_i}(\hat{\boldsymbol{w}}_i^{k,E} - \boldsymbol{w}^{k,0}) - \boldsymbol{w}^*\|^2$
$= \|\boldsymbol{w}^{k,0} + \sum_{i=1}^N \frac{\alpha_i}{1-CER_i}(\hat{\boldsymbol{w}}^{k,i} - \boldsymbol{w}^{k,0}) - \boldsymbol{w}_{\text{id}}^{k+1,0}\|^2 + \|\boldsymbol{w}_{\text{id}}^{k+1,0} - \boldsymbol{w}^*\|^2$
$= \|\sum_{i=1}^N \frac{\alpha_i}{1-CER_i}(\hat{\boldsymbol{w}}_i^{k,E} - \boldsymbol{w}^{k,0}) - \alpha_i(\boldsymbol{w}_i^{k,E} - \boldsymbol{w}^{k,0})\|^2 + \|\boldsymbol{w}_{\text{id}}^{k+1,0} - \boldsymbol{w}^*\|^2$
$\leq \sum_{i=1}^N \alpha_i\sum_{l=0}^d \|\frac{[\hat{\boldsymbol{w}}_i^{k,E} - \boldsymbol{w}^{k,0}]_l}{1-CER_i} - [\boldsymbol{w}_i^{k,E} - \boldsymbol{w}^{k,0}]_l\|^2 + \|\boldsymbol{w}_{\text{id}}^{k+1,0} - \boldsymbol{w}^*\|^2$, where we applied, in order, the properties of the projection, the definition of $\boldsymbol{w}_{\text{id}}^{k+1,0}$, the triangle inequality, and the Jensen's inequality.

Computing the expectation over random events at round k, we obtain: $\mathbb{E}_{B^k,T^k}\|\boldsymbol{w}^{k+1,0} - \boldsymbol{w}^*\|^2 \leq \sum_{i=1}^N \alpha_i\mathbb{E}_{B^k}\|\boldsymbol{w}_i^{k,E} - \boldsymbol{w}^{k,0}\|^2 \cdot \Big[(1-CER_i)(\frac{1}{1-CER_i} - 1)^2 + CER_i\Big] + \mathbb{E}_{B^k}\|\boldsymbol{w}_{\text{id}}^{k+1,0} - \boldsymbol{w}^*\|^2 = \sum_{i=1}^N \alpha_i\frac{CER_i}{1-CER_i}\mathbb{E}_{B^k}\|\boldsymbol{w}_i^{k,E} - \boldsymbol{w}^{k,0}\|^2 + \mathbb{E}_{B^k}\|\boldsymbol{w}_{\text{id}}^{k+1,0} - \boldsymbol{w}^*\|^2$.

Moreover,
$\mathbb{E}_{B^k}\|\boldsymbol{w}_i^{k,E} - \boldsymbol{w}^{k,0}\|^2 = \mathbb{E}_{B^k}\|\sum_{e=0}^{E-1}(\eta^k)^2\nabla f_i(\boldsymbol{w}_i^{k,l}, B_i^{k,l})\|^2$
$\leq E\sum_{e=0}^{E-1}(\eta^k)^2\,\mathbb{E}_{B^k}\|\nabla f_i(\boldsymbol{w}_i^{k,l}, B_i^{k,l})\|^2 \leq (\eta^k)^2E^2G^2$.
We conclude by combining the two inequalities above. □

Let $A = E^2G^2\sum_{i=1}^N \alpha_i\frac{CER_i}{1-CER_i} + 4LE(1+LE)\Gamma + E^2\sigma^2 + \frac{1}{2}E^2\sigma^2 + \frac{1}{6}E(E-1)(2E-1)G^2$.
By combining Lemmas 1–4, we obtain: $\frac{E}{2}\eta^k\sum_{i=1}^N\Big(f_i(\boldsymbol{w}^{k,0}) - f_i(\boldsymbol{w}^*)\Big) \leq \|\boldsymbol{w}^{k,0} - \boldsymbol{w}^*\|^2 - \mathbb{E}_{B^k,T^k}\|\boldsymbol{w}^{k+1,0} - \boldsymbol{w}^*\|^2 + (\eta^k)^2A$.

Summing for $k = 1, \ldots, K$, and computing the expectation over the whole history, we obtain:
$\frac{E}{2}\sum_{k=1}^K \eta^k\sum_{i=1}^N \mathbb{E}\Big[f_i(\boldsymbol{w}^{k,0}) - f_i(\boldsymbol{w}^*)\Big]$
$\leq \|\boldsymbol{w}^{1,0} - \boldsymbol{w}^*\|^2 - \mathbb{E}\|\boldsymbol{w}^{k+1,0} - \boldsymbol{w}^*\|^2 + A\sum_{k=1}^K(\eta^k)^2$.

The result follows by dividing both terms by $\frac{E}{2}\sum_{k=1}^K \eta^k$ and observing that, by convexity of the functions f_i:
$\mathbb{E}\Big[F(\overline{\boldsymbol{w}}^{k,0}) - F(\boldsymbol{w}^*)\Big] = \mathbb{E}\Big[\sum_{i=1}^N \alpha_i\left(f_i(\overline{\boldsymbol{w}}^k) - f_i(\boldsymbol{w}^*)\right)\Big]$
$\leq \sum_{i=1}^N \alpha_i\frac{\sum_{k=1}^K \eta^k\mathbb{E}\Big[f_i(\boldsymbol{w}_i^{k,0}) - f_i(\boldsymbol{w}^*)\Big]}{\sum_{k'=1}^K \eta^{k'}}$.

REFERENCES

[1] H.-H. Chang *et al.*, "Distributive Dynamic Spectrum Access through Deep Reinforcement Learning: A Reservoir Computing Based Approach," *IEEE Internet of Things Journal*, 2018.

[2] S. D'Oro *et al.*, "Can you fix my neural network? real-time adaptive waveform synthesis for resilient wireless signal classification," *Proc. of IEEE Intl. Conf. on Computer Communications (INFOCOM)*, May 2021.

[3] D. Uvaydov *et al.*, "DeepSense: Fast wideband spectrum sensing through real-time in-the-loop deep learning," *Proc. of IEEE Intl. Conf. on Computer Communications (INFOCOM)*, May 2021.

[4] C. Liu *et al.*, "Deep CM-CNN for Spectrum Sensing in Cognitive Radio," *IEEE Journal on Selected Areas in Communications*, vol. 37, no. 10, pp. 2306–2321, 2019.

[5] A. Al-Shawabka *et al.*, "Exposing the Fingerprint: Dissecting the Impact of the Wireless Channel on Radio Fingerprinting," *Proc. of IEEE Conference on Computer Communications (INFOCOM)*, 2020.

[6] F. Restuccia *et al.*, "DeepRadioID: Real-Time Channel-Resilient Optimization of Deep Learning-based Radio Fingerprinting Algorithms," *Proc. of ACM International Symposium on Mobile Ad Hoc Networking and Computing (MobiHoc)*, 2019.

[7] F. Adelantado *et al.*, "Understanding the Limits of LoRaWAN," *IEEE Communications Magazine*, vol. 55, no. 9, pp. 34–40, 2017.

[8] J. Konečnỳ *et al.*, "Federated optimization: Distributed optimization beyond the datacenter," *arXiv preprint arXiv:1511.03575*, 2015.

[9] M. M. Amiri *et al.*, "Federated Learning over Wireless Fading Channels," *IEEE Transactions on Wireless Communications*, vol. 19, no. 5, pp. 3546–3557, 2020.

[10] S. Niknam *et al.*, "Federated Learning for Wireless Communications: Motivation, Opportunities, and Challenges," *IEEE Communications Magazine*, vol. 58, no. 6, pp. 46–51, 2020.

[11] K. Yang *et al.*, "Federated Learning via Over-the-air Computation," *IEEE Transactions on Wireless Communications*, vol. 19, no. 3, pp. 2022–2035, 2020.

[12] B. Su *et al.*, "Energy Efficient Resource Allocation for Uplink LoRa Networks," in *2018 IEEE Global Communications Conference (GLOBECOM)*, pp. 1–7, 2018.

[13] X. Liu *et al.*, "Resource Allocation in Wireless Powered IoT Networks," *IEEE Internet of Things Journal*, vol. 6, no. 3, pp. 4935–4945, 2019.

[14] R. Jin *et al.*, "On the design of communication efficient federated learning over wireless networks," 2020.

[15] V. D. Nguyen *et al.*, "Efficient Federated Learning Algorithm for Resource Allocation in Wireless IoT Networks," *IEEE Internet of Things Journal*, pp. 1–1, 2020.

[16] L. Bonati *et al.*, "Colosseum: Large-scale wireless experimentation through hardware-in-the-loop network emulation," in *2021 IEEE International Symposium on Dynamic Spectrum Access Networks (DySPAN)*, pp. 105–113, IEEE, 2021.

[17] J. Haxhibeqiri *et al.*, "A survey of LoRaWAN for IoT: From technology to application," *Sensors*, vol. 18, no. 11, p. 3995, 2018.

[18] B. McMahan *et al.*, "Communication-efficient learning of deep networks from decentralized data," in *Artificial Intelligence and Statistics*, pp. 1273–1282, PMLR, 2017.

[19] J. Wang *et al.*, "A field guide to federated optimization," *arXiv preprint arXiv:2107.06917*, 2021.

[20] A. Rodio *et al.*, "Federated learning under heterogeneous and correlated client availability," in *IEEE Intl Conf. on Computer Communications (INFOCOM)*, 2023.

[21] X. Li *et al.*, "On the convergence of FedAvg on non-iid data," *arXiv preprint arXiv:1907.02189*, 2019.

[22] D. Croce *et al.*, "LoRa Technology Demystified: From Link Behavior to Cell-Level Performance," *IEEE Transactions on Wireless Communications*, vol. 19, no. 2, pp. 822–834, 2020.

[23] Semtech Corporation, "LoRa™ Modulation Basics," tech. rep., Camarillo, CA, USA, 2015.

[24] Semtech Corporation, "Predicting LoRaWAN Capacity," may 2020.

[25] W. Xing *et al.*, "Parallel machine scheduling with splitting jobs," *Discrete Applied Mathematics*, vol. 103, no. 1-3, pp. 259–269, 2000.

[26] F. Busacca *et al.*, "SDR-LoRa: dissecting and implementing LoRa on Software-Defined Radios to advance experimental IoT research," in *Proceedings of the 16th ACM Workshop on Wireless Network Testbeds, Experimental evaluation & CHaracterization*, pp. 24–31, 2022.