IEEE Access
Multidisciplinary ⋮ Rapid Review ⋮ Open Access Journal

Received 1 July 2023, accepted 21 July 2023, date of publication 28 July 2023, date of current version 7 August 2023.

Digital Object Identifier 10.1109/ACCESS.2023.3299443

RESEARCH ARTICLE

A Discrete-Time Multi-Hop Consensus Protocol for Decentralized Federated Learning

DANILO MENEGATTI[1], ALESSANDRO GIUSEPPI[1], (Senior Member, IEEE), SABATO MANFREDI[2], AND ANTONIO PIETRABISSA[1], (Senior Member, IEEE)

[1]Department of Computer, Control, and Management Engineering, University of Rome "La Sapienza," 00185 Rome, Italy
[2]Department of Electrical Engineering and Information Technology, University of Naples Federico II, 80125 Naples, Italy

Corresponding author: Danilo Menegatti (menegatti@diag.uniroma1.it)

This work has been partially carried out in the framework of the CADUCEO project (No. F/180025/01-05/X43), supported by the Italian Ministry of Enterprises and Made in Italy, and in part in the scope of the project FedMedAI, POR FESR Lazio 2014–2020 (Azione 1.2.1), funded by the Lazio region under Grant A0375-2020-36491-23/10/2020.

ABSTRACT This paper presents a Federated Learning (FL) algorithm that allows the decentralization of all FL solutions that employ a model-averaging procedure. The proposed algorithm proves to be capable of attaining faster convergence rates and no performance loss against the starting centralized FL implementation with a reduced communication overhead compared to existing consensus-based and centralized solutions. To this end, a Multi-Hop consensus protocol, originally presented in the scope of dynamical system consensus theory, leveraging on standard Lyapunov stability discussions, has been proposed to assure that all federation clients share the same average model employing only information obtained from their m-step neighbours. Experimental results on different communication topologies and the MNIST and MedMNIST v2 datasets validate the algorithm properties demonstrating a performance drop, compared with centralized FL setting, of about 1%.

INDEX TERMS Discrete-time consensus, multi-hop, federated learning, distributed systems.

I. INTRODUCTION

Consensus theory, a branch of dynamical system theory that deals with the coordination of distributed systems, has been recently investigated in the context of Federated Learning (FL), an emerging research trend for distributed machine learning [1]. FL was designed as a solution for machine learning applications in which data could not be shared, collected or re-distributed in any way, hence imposing the requirement of having distributed learning agents cooperating to solve a complex task in a privacy-preserving way [2]. The typical FL algorithms envisage the presence of a centralized entity that coordinates the learning by overseeing a model averaging procedure [1], and finds more and more application in mutiple scenarios, such as the medical one [3]. Nevertheless, recently, fully decentralized approaches for FL have been investigated. In [4] a consensus-based FL algorithm leveraging on mutual cooperation of edge devices in wireless Internet of Things (IoT) networks is proposed, whereas in [5] a similar algorithm have been developed based on first-order dynamic average consensus, while in [6] the authors proposed a FL protocol based on discrete-time average consensus.

This work proposes the inclusion of a so-called multi-hop consensus protocol in the design of a FL algorithm, extending the previously proposed FedLCon algorithm [6] with a new consensus law to mitigate its main limitations. The main focus objective of the proposed algorithm is to offer a solution that may be used to decentralize, with virtually no performance loss, any FL application that relies on a model averaging procedure. The specific multi-hop protocol employed in this paper was first introduced in [7] as a solution to achieve faster consensus by exploiting multiple-hop paths in the network. Contrary to the standard consensus setting, in which the agents exchange information only with their topological neighbours, multi-hop envisages this exchange to also include information regarding a collection of

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

their neighbours. The main characteristics of the proposed FL algorithm are:

- The significant reduction (of about 66%) of the number of communication rounds needed to assure convergence compared to the previously proposed, 1-hop, algorithm FedLCon.
- The complete transparency of the consensus protocol with respect to the model averaging procedure that characterizes FL, which makes the proposed solution seamlessly deployable to the most recent horizontal FL algorithms.
- The mathematical assurance that the decentralization does not cause any significant performance degradation in terms of the quality of the trained models, attained through the assurance of implementing an exact distributed version of the model averaging procedure.

The remainder of the paper is organized as follows: Section II discusses the relevant works in the literature; Section III provides the reader with some background on FL and discrete-time weighted consensus theory; Section IV details the proposed multi-hop protocol; Section V shows the applicability of the proposed solution to two test scenarios; Section VI draws the conclusions and presents possible future works.

II. RELATED WORKS

A common problem when dealing with multi-agent systems (MAS) is the so-called ''group agreement'' that is the consensus problem. Given a distributed MAS, each agent, which is in principle a dynamical system, seeks to agree with the others upon certain quantities of interest by using only a limited amount of information received from the other agents [8].

Due to its very general formulation, the study of the consensus problem has been conducted across multiple fields, such as computer science [9], leading to the foundation of distributed computing, management science and statistics [10], [11], [12], and system and control theory [13], [14], [15], [16].

At its core, FL can be seen as a group agreement problem in which a machine learning model, such as a deep neural network, has to be trained in a distributed way relying on the cooperation of a set of agents/clients. Ever since its original formulation [1], the fundamental characteristic of any FL algorithm is to be found its ability to deal with a non-independently and identically distributed (non-IID) and imbalanced data distribution *as-it-is*, meaning that in a FL setting any collection and/or re-distribution of the training data is avoided, typically for privacy/confidentiality constraints. This capability is attained through a so-called model averaging procedure, according to which a centralized server periodically collects the locally trained machine learning to evaluate a federation-wise average model by averaging the models' parameters. Over the past few years, a significant research effort has been spent on developing new FL solutions, improving its capabilities with extensions mainly focused in two directions:

- enhancing the communication between the server and the clients, for example by reducing the number of parameters transmitted [17], [18], [19] or the number of clients taking part in the averaging procedure, [20] or by implementing further privacy-assuring measures such as weight/gradient encryption [21] or data encryption for encrypted training [22];
- engineering the training loss function to attain better performance with fewer rounds of communications, as done by the FedProx [23], [24], FedDM [19] and FedSR [25] algorithms.

Compared to the original setting, which involved applications typically related to edge/IoT devices connecting to a central server, FL has evolved to tackle new application domains by the integration with technologies from other computer science fields such as cybersecurity, cryptography and blockchain, leading to algorithms that enable reliable and trustworthy FL applications robust against malicious clients [26], [27], [28] , corrupted servers [29] and unreliable communications [30].

As mentioned, the present study aims to design an efficient solution to allow the complete decentralization of other existing and future FL algorithms as well as to achieve faster convergence rates, removing the typical requirement of having a centralized entity that governs the model averaging process. In this direction, we rely on results from consensus theory, and in particular from the so-called average consensus field [31], to design an information exchange protocol under which the federation is able to exactly decentralize a centralized model averaging. Due to the exact nature of the decentralization, and its independence from the training process and its loss, the proposed solution is in principle applicable to any of the mentioned FL algorithms, which instead modify mainly communication-related aspects or the characteristics/objectives of the local trainings. For the sake of presentation clarity, we will present the developed algorithm, based on multi-hop consensus, applied to the original FedAvg.

The theoretical foundations for addressing the consensus problem have been explored in works such as [32] and [33], where a dynamical system is used both to model the information exchanges between vehicles and to let them achieve consensus in the context of linear time-invariant (LTI) systems with fixed time delay. Works such as [34] have studied linear and nonlinear consensus protocols for networked dynamic systems subject to a fixed communication topology, while [8] extended such results to switching/time-varying topologies. More general cases and properties, including time-delays and robustness, have been investigated in [35].

From an application perspective, consensus has been studied and deployed in multiple MAS domains. Among such domains, one of the most popular application has been related to formation control. In works such as [36], inter-vehicle

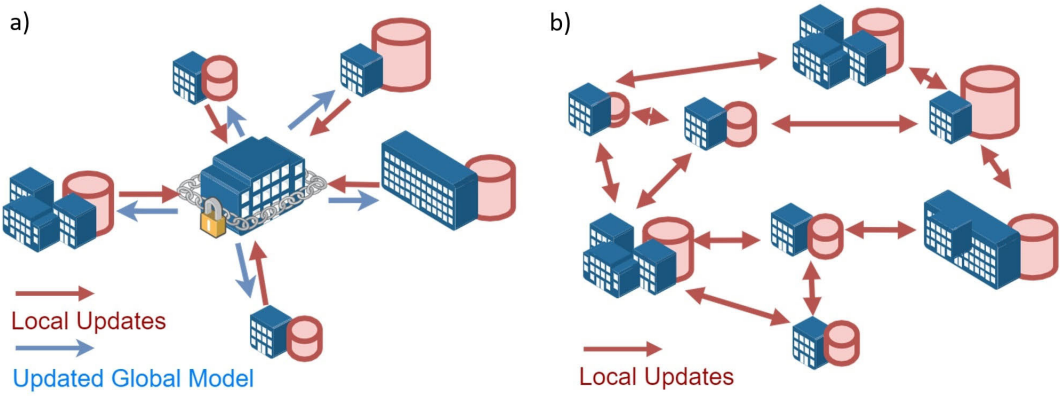

FIGURE 1. (a) Standard Federated Learning system with a secure server (left) [1]; (b) Example of a decentralized FL setting with bi-directional point-to-point communications (right).

interconnections are modeled as a graph, whose Laplacian properties are studied to infer stability of the considered formation, while in [37] a local collision-free stabilizing nonlinear state feedback for the formation is derived using potential functions [38] and a graph modelling interconnections. In [39], a control law for a group of unicycles connected by a communication network is derived via artificial potential fields and consensus algorithms, allowing obstacle avoidance and rendezvous stability.

Consensus theory has also been broadly applied to smart networks, among which we mention smart power systems. In [40] a consensus-based droop control for real and reactive power sharing in microgrids is proposed, while [41] applies s consensus protocol to synchronize the state-of-charge (SoC) and power levels of batteries.

Multi-hop techniques are commonly adopted in the areas of computer science [9], [42] and have found application also for consensus problems. In [7] a multi-hop relay technique is introduced to increase the convergence speed to the consensus equilibrium, while in [43] a multi-hop-based consensus protocol is employed to solve the global leader-follower consensus problem. Moreover, application of multi-hop communication in wireless sensor networks have been investigated from a control control perspective in [44].

As mentioned, dynamical system consensus-based algorithms have been employed also in the context of FL, as in [4], [5], and [6]. Centralized FL approaches, as depicted in Figure 1 (a), involve a server that collects the DeepNN trained by each of the federation clients on the basis of their private data. After this collection is conducted, the server performs a model averaging procedure according to which an average DeepNN is obtained from the various DeepNN collected and is then propagated back into the federation for further distributed training.

Several decentralized FL studies consider a federation characterized by a sparse communication topology in which each client acts as a server for its neighbours, as shown in Figure 1 (b. In particular, the authors of [4] propose a direct consensus-based extension of FedAvg [1] called *Consensus-based Federated Averaging* CFA, and its extension *Consensus based Federated Averaging with Gradient Exchange* CFA-GE which improves the speed of the learning procedure by allowing each agent to exploit training gradients coming from its neighbours. The work [5] introduces a FL algorithm based on the first order dynamic average consensus algorithm (FODAC) [45] applied to both time-invariant and time-varying topologies. All these works decentralize the FL process implementing a single iteration of a consensus protocol between the local training steps, implying that the consensus and training procedure are linked and one may influence the convergence of the other. On the contrary, as mentioned the present study assures that at the beginning of each training round every federation client shares the same average set of parameters, which also correspond to parameters that a centralized server would have computed in a centralized FL solution. This characteristic allows for a stronger convergence assurance and expected performance, as the learning process becomes transparent with respect to the consensus procedure, meaning that, from the learning clients' perspective, the system evolves as if the centralized server was in place.

Distributed FL approaches, not based on consensus, have been explored in works such as [46], where a hierarchical architecture is employed to allow for massive federations in which multiple local servers cooperate in a distributed fashion, [47] that relies on Bayesian learning for device-to-device (D2D) networks and [48], that, similarly to this work, focuses on improving the consistency/similarity of the client's models in a fully distributed FL setting. A different approach has been followed by the authors in [49], where they presented a centralized solution enhanced by client-to-client communications. The present paper, and works such as [30] and [50], rely on graph/matrix theory to characterize the communication topology of the federation and enhance the capabilities to allow its deployment for new applications.

The above-mentioned works, as the present paper, focus on a so-called Horizontal Federated Learning (HFL) setting, in which all clientts of the federation share a common model architecture and seek to reach a consensus on its weights or, equivalently, the values of the optimal weights for the average client. On the contrary, Personalized Federated Learning (PFL) [51], [52] is a recent branch of FL that investigates settings in which each client requires a fine-tuned, individualized, model.

III. PROBLEM FORMULATION

Consider a group, or federation, I of N cooperating agents and Suppose that each client in I stores data, which is of the same type for all nodes, and does have computational capability.

FL aims to solve a distributed machine learning (ML) problem in a cooperative way, making use of the above mentioned computational capability and the data of each clients, without requiring the exchange of any data among the clients.

The standard FL setting envisages that all clients in the federation share a common ML model architecture (e.g., the layers and structure of a Deep Neural Network (DeepNN)) but have no direct means of training such model on the totality of the federation data, as every client only possess a small portion of data that it cannot share with the other cleints.

In other words, a client $i \in I$ may only train its DeepNN using its own dataset $D_i = \{(\alpha_n, \beta_n), \quad n \in \{1, \ldots, N\}\}$, with α_n and β_n being an input sample and its corresponding label/ground-truth value.

Such training process consist, as customary in ML, in the iterative update, in a gradient-descent fashion, of the ML model parameters/weights w_i, by means of the minimization of a so-called ''loss function''

$$L_i = \sum_{(\alpha_n, \beta_n) \in D_i)} l_i((\alpha_n, \beta_n)|w_i), \tag{1}$$

where the function l_i captures some quantity whose minimization is related to solving the ML problem as, for example, the mean squared error between the DeepNN guest and a label in regression tasks or the categorical cross-entropy in classification problems.

Note that, since the DeepNN architeture is common among all the clients, the cardinality of all the weights vectors is the same, i.e., $|w_i| = |w_j|$ for all $i, j \in I$, but since the various datasets D_i are different, $w_i \neq w_j$ for all $i, j \in I$.

The goal of the federation is then to identify the set of weights w^* that minimizes the joint cost function [53], defined as

$$\mathcal{L}(w) := \sum_{i \in I} p_i L_i(w), \tag{2}$$

with $p_i = \mid D_i \mid / \mid \mathbb{D} \mid$, where $\mathbb{D} = \bigcup_i D_i$ denotes the total available data, and $L_i(w)$, the loss function of client i over its entire dataset D_i, is defined as

$$L_i(w) = L_i(D_i \mid w) = \frac{1}{\mid D_i \mid} \sum_{(\alpha_n, \beta_n) \in D_i} l_i((\alpha_n, \beta_n) \mid w). \tag{3}$$

In other words, the goal of the training procedure is to find the optimal vector of parameters w^* which, if shared among all the clients of the federation, minimizes (2).

While in a *standard ML scenario* the above mentioned minimization of (2) is solved in a centralized fashion, by means of a computing entity that has ccess to the entirety of the available data, in a *distributed setting* data are arbitrarily distributed over the federation, implying that the clients are required to cooperate [1]. From a mathematical perspective, this cooperation implies that the gradient $\nabla\mathcal{L}(w)$ has to be estimated from the gradients $\nabla L_i(w_i)$ evaluated by the clients.

In order to solve the problem, if one could assume that the data D_i owned by the various clients are independent and identically distributed (IID) with respect to $\mathbb{D}$, and hence $\mathbb{E}_{D_i}[L_i(w)] = \mathcal{L}(w) \forall i \in I$, it would follow that $L_i(w)$ provides a good approximation of $\mathcal{L}(w)$ and the locally computed gradients $\nabla L_i(w_i)$ could be averaged to reconstruct $\nabla L(w)$. This approach is the one typically employed in distributed settings, in which a centralized entity partitions the data so that the IID hypothesis holds as common in multi-datacenter and parallel ML solutions [1].

However, in a *federated scenario*, nothing can be said about the data of each client, which is processed without any redistribution, and therefore, such IID hypothesis cannot be assumed, i.e., $L_i(w)$ could provide an arbitrarily bad approximation of $\mathcal{L}(w)$. For this particular reason, in [1] the authors presented a round-based iterative procedure for model averaging, called Federated Averaging (FedAvg).

Given a centralized server which orchestrates the learning procedure of the federation of clients, FedAvg is based on two main steps, i) *local training* and ii) *centralized averaging*, both iteratively repeated.

In the first phase, the server chooses a subset of clients which undergo local training over their own dataset D_i, while in the second phase the server performs a weighted average of the clients' weights w_i it collected and then forwards the averaged weights to the entirety of the federation.

Algorithm 1 reports the presudo-code of FedAvg.

Even if several variants of FL algorithms have been developed (e.g., [54]), most of the solutions available in the literature share with FedAvg both the centralized setting and the two-step iterative structure of the training process.

In addition to centralized variants, several decentralized algorithms have also been developed, in order to overcome the single point of failure vulnerability related to the requirement of a learning-orchestrating centralized server, whose behaviour needs to be completely trusted by all the clients.

In general, decentralized FL deal with federation characterized by an arbitrary communication topology that can be modeled by an undirected graph $G = (I, E)$, consisting of the set I of N clients and the set E of edges, which model the possibility of exchanging information between two

Algorithm 1 FedAvg [1]

SERVER UPDATE
for each communication round $t = 1, \ldots, T$ **do**
  select a subset of clients for the averaging procedure
  for all selected client i **do**
    CLIENT UPDATE
    Receive $\tilde{w}_i$ from client i
  end for
  set $w(t) = \sum_i p_i \tilde{w}_i(t)$
  propagate $w(t)$ in the federation $(w_i(t) = w(t), \forall i)$
end for

CLIENT UPDATE
for each local epoch e from 1 to E **do**
  for each mini-batch b from D_i **do**
    $w_i(t-1) \leftarrow w_i(t-1) - \eta \nabla L_i(b \mid w(t-1)),\ 0 < \eta < 1$
  end for
end for
set $\tilde{w}_i(t) = w_i(t-1)$
return $\tilde{w}_i(t)$ to the server

clients. In this framework, an immediate decentralization of FedAvg, named DecFedAvg, was proposed in [6], envisaging the model averaging procedure to be conducted by every client on the basis of the weights collected from the its neighbours. The signficant shortcoming of such a solution is that, contrary to what happens in the centralized setting with FedAvg, after each model averaging procedure, the various clients i do not share the same set of weights and hence the performance of the various DeepNN may vary significantly over the federation.

To overcome this limitation, a consensus-based algorithm, FedLCon, was also proposed in [6]. According to this algorithm, the model averaging procedure is replaced by a so-called "consensus round" that requires the clients to exchange their weights according to a standard discrete-time weighted-average consensus law. At the end of every consensus round, the various clients share the same weights, effectively decentralizing FedAvg without any performance loss. In order to archive this result, a significant communication overhead is introduced, as each consensus round requires a number of information exchanges that depends on the federation topology and in particular on its connectivity. To reduce the impact of this overhead, the next section will present an extension of FedLCon that employs a multi-hop consensus strategy.

Remark 1: Like FedAvg [1], also FedLCon [6] and its extension to multi-hop protocols proposed here are completely transparent to the particular DeepNN employed to solve the considered machine learning task. In other words, the proposed solution will work without any modification regardless of the neural network architecture being employed.

IV. MULTI-HOP

Let $G = (I, E)$ be an undirected graph used to represent the communication topology of a networked-MAS, with I being the set of N clients, and $E \subseteq I^2$ being the set of edges; if there exists an edge (n_i, n_j) between clients n_i and n_j, it means that n_j can receive information from n_i.

A path is defined as a sequence of distinct clients $[n_1, n_2, \ldots, n_k]$, such that $(n_{i-1}, n_i) \in E$, with $i = 1, \ldots, k$, and it is called an *m-hop* path if it includes $m + 1$ clients. An undirected graph is defined as connected if there exist a path between any two nodes.

It is now possible to define the following matrices [55]: The adjacency matrix $A = \{a_{ij}\} \in R^{N \times N}$ is defined as

$$a_{ij} = \begin{cases} 1, & (n_i, n_j) \in E \\ 0, & \text{otherwise} \end{cases}$$

The degree matrix $D = d_{ij} \in R^{N \times N}$, where $d_{ij} = \sum_j a_{ij}$, is a diagonal matrix and each diagonal entry corresponds to the degree of node i, namely to the node's number of edges. The Laplacian matrix is then $L = D - A$.

Let $x_i(t)$ be the state of node n_i at time t, and $N_i = \{n_j \in I : (n_i, n_j) \in E\}$ be the set of its neighbours; a networked-MAS is said to achieve *consensus* at a certain time $\bar{t}$ if $x_i(\bar{t}) = x_j(\bar{t})$, for all n_i and $n_j \in I$.

It is well known that under the hypothesis of a connected undirected graph, and under the following consensus protocol

$$\dot{x}_i(t) = \sum_{j \in N_i} a_{ij} \left(x_j(t) - x_i(t)\right) \tag{4}$$

the networked-MAS reaches consensus, and each node share a common *consensus value*

$$\bar{x} = \frac{1}{N} \sum_i x_i(0) \tag{5}$$

If each node $n_i \in I$ is associated with a weight p_i, it is possible to defined the weight matrix $P = diag(p_i) \in R^{N \times N}$, and consider the *weighted-average consensus* problem. Under the same assumption on the connectivity of the graph of the previous case, following the following consensus protocol

$$p_i \dot{x}_i(t) = \sum_{j \in N_i} a_{ij} \left(x_j(t) - x_i(t)\right) \tag{6}$$

the system reaches a consensus value that coincides with the weighted average of their initial conditions:

$$\bar{x} = \frac{\sum_{i \in I} p_i x_i(0)}{\sum_{i \in I} p_i}. \tag{7}$$

Extending the previous considerations to discrete time, assuming that the sampling time ϵ is such that $\epsilon < \min_{i \in I}(p_i/d_{ii})$ [8], [31], the law (6) can be discretized as

$$x_i(t+1) = x_i(t) + \frac{\epsilon}{p_i} \sum_{j \in N_i} a_{ij}(x_j(t) - x_i(t)). \tag{8}$$

that in matrix form [31] becomes

$$x_i(t+1) = H_p x(t), \tag{9}$$

with $H_p = I_N - \epsilon P^{-1} L$, where I_N is the identity matrix of size N.

From (9), starting from the well-known definition of dominant time constant for a discrete-time linear time-invariant system and its settling time [56], it follows that the agents will reach convergence, with precision of 99%, after a number of steps n_ϵ:

$$n_\epsilon = 5 \max_{i \in I} \left\lceil \frac{-1}{\ln(|\ \lambda_i(H_p)\ |)} \right\rceil, \tag{10}$$

where $\lambda_i(H_p)$ is the i-th eigenvalue different from 1 of the matrix H_p and $\lceil \cdot \rceil$ denotes the ceiling function of its argument, with a resulting 1%-settling time $t_a \approx n_\epsilon \cdot \epsilon$.

The consensus framework can be employed in the FL by providing an appropriate interpretation to the clients of a given federation. Even if they are not dynamical system in the strict sense of the term, the weights $w_i(t)$ of the DeepNN of each client can be seen as its state $x_i(t)$ which evolves in time as FL training procedure goes on. Having the goal of learning a high-performance machine learning model specific to the particular task being addressed, i.e., image segmentation, classification, regression, without sharing their own data, the clients, by iteratively updating their weights as described in [1], can be seen as a group of agents seeking consensus.

As a result, while FedAvg requires a centralized server to orchestrate the entire learning procedure, a consensus-based approach such as FedLCon [6] provides a fully decentralized solution. In fact, the weights of each client are updated via a *consensus round* which lasts n_ϵ (10) times; by setting their initial states equal to their weights resulting from the learning procedure, i.e., using the symbols of Algorithm 1, $x_i(0) = \tilde{w}_i(t)$, for all $i \in I$, equation (8) becomes

$$x_i(k+1) = x_i(k) + \frac{\epsilon}{|D_i|} \sum_{j \in N_i} a_{ij} \left(x_j(k) - x_i(k) \right) \tag{11}$$

where ϵ and D_i are defined as before, and the corresponding *consensus value* becomes

$$x_i(n_\epsilon) \approx \frac{\sum_i |D_i| \tilde{w}_i(t)}{|\mathbb{D}|}, \quad \forall i \in I \tag{12}$$

meaning that, after n_ϵ consensus iterations, the state of each client corresponds to an approximation of the weights computed in a centralized way in FedAvg - see line 8 of Algorithm 1. The procedure is iteratively repeated whenever communication among clients is permitted. Algorithm 2 reports the pseudo-code for FedLCon.

As the consensus procedure is transparent to any FedAvg-like algorithm, the present work allows the use of multi-hop communications within the FedLCon algorithm, providing each client with more information to update its weights, coming not only from its neighbours, i.e., 1-hop communication, but also from the neighbours of its neighbours, i.e., m-hop communications.

Algorithm 2 Multi-Hop FedLCon

DECENTRALISED FEDERATED LEARNING
for all communication rounds $t = 1, \ldots, T$ **do**
  for all clients $i \in I$ **do**
    for each local epoch e from 1 to E **do**
      for each mini-batch b from D_i **do**
        $w_i(t-1) \leftarrow w_i(t-1) - \eta \nabla L_i(b | w_i(t-1))$
      end for
    end for
    set $\tilde{w}_i(t) = w_i(t-1)$
  end for
  update $w_i(t)$ via a CONSENSUS ROUND
end for

CONSENSUS ROUND
Compute n_ϵ according to (10) depending on the topology
Set $x_i(0) = \tilde{w}_i(t)$ for all clients $i \in I$
for $k = 0, \ldots, n_\epsilon - 1$ **do**
  for all clients $i \in I$ **do**
    update x_i according to (14)
  end for
end for
set $w_i(t) = x_i(n_\epsilon)$ for all clients $i \in I$

Given the graph G, suppose to consider two-hop paths; the corresponding consensus protocol [7] is described as follows

$$\dot{x}_i = -\sum_{j \in N_i} a_{ij} \left(\left(x_i - x_j \right) + \sum_{k \in N_j} a_{jk} \left(x_i - x_k \right) \right) \tag{13}$$

As a result, client i expands its knowledge by receiving not only information coming from its neighbours j, but also from the neighbours of j resulting in a better convergence performance of the network. The approach of multi-hop may be extended to m-hop relay case. Therefore the multi-hop relay protocol can improve the convergence speed without increasing the number of links and cost, but at the price of extra communication bandwidth. In addition, with the increasing of the more number of hop m the protocol leverages, the faster the convergence speed is, while the more sensitive the protocol is to the time delay. This arises a trade-off among convergence speed, bandwidth and sensitivity to time delays to account in the protocol design [57]. In this respect the 2-hop relay protocol strongly improves convergence speed with a reduced detriment of delay-stability margin and bandwidth. This is why for the considered healthcare application, that usually presents low-latency communication, we will consider $m = 2$. Moreover, the extra requirement of bandwidth may be mitigated by transmitting in the same packet the information of the state of both a node and its neighbors, at a price of a small reduction of delay-stability margin. This also reduces packets lost for collision phenomena, making this algorithm appealing also for edge-computing scenarios [57].

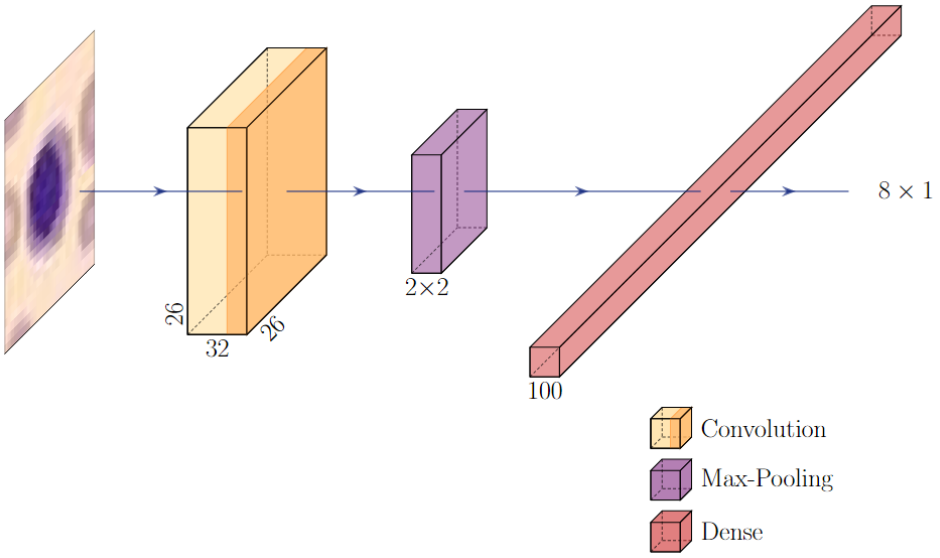

FIGURE 2. Test DeepNN architecture.

Starting from G and allowing for two-hop communications, it is possible to define the *two-hop undirected graph* $\hat{G} = (I, \hat{E})$, where $\hat{E}$ corresponds to the two-hop paths of G. Finally, the joint undirected graph $\tilde{G} = G \bigcup \hat{G} = (I, E \bigcup \hat{E})$.

Applying the above mentioned consensus protocol to FedLCon is quite straightforward, as the only modification taken into account is related to equation (11), which becomes

$$x_i(k+1) = x_i(k) + \frac{\epsilon}{|D_i|} \sum_{j \in N_i} a_{ij} \Big(\big(x_j(k) - x_i(k)\big) + \sum_{k \in N_j} a_{jk} \left(x_i(k) - x_k(k)\right) \Big), \tag{14}$$

whose convergence is assured by selecting n_ϵ according to (10) using the matrices that characterized the m-hop communication graph $\hat{G}$.

V. CASE STUDY

A. SETUP

In this section, we do present a comparison of multi-hop communications within the FedLCon algorithm over the MNIST [58] and MedMNIST v2 [59] datasets in terms of performance and comunication overhead.

In both cases we do solve a multi-class classification problem; with regards to the MNIST dataset, being a collection of handwritten digits between 0 and 9, a DeepNN is trained to discriminate among nine classes; while with respect to the MedMNIST v2 dataset, our testing was conducted on its sub-set BloodMNIST [60] that consists of an eight-classes classification problem for microscopic peripheral blood cell images.

The considered DeepNN shares the same architecture for both scenarios; the output of a 2D convolutional layer of 32 filters and kernel size (3,3) undergoes a pooling operation of size (2,2) before being flattened and given to one dense layer with 100 neurons and ReLu [61] activation function. A graphical representation of the DeepNN is reported in Figure 2. Being the proposed algorithm completely transparent to the particular type of neural network architecture under consideration, it was decided to a compromise between architecture complexity and performance. Note that the number of trainable parameters in the DeepNN does not impact the consensus protocol, as (14) implies that each component of the vectors x_i does not depend on the others (i.e., each DeepNN parameter converges towards its average independently from the others).

We set the parameters of the algorithm as local training epochs $E = 2$, mini-batch size $b = 32$, communication rounds $T = 15$. Unlike the number of communication rounds which is arbitrarily chosen, the choice of a small number of local training epochs is mainly related to the limited computational capabilities of the clients of the considered federation: since no assumption has been made regarding the type of clients, we do make a conservative choice. The simulations were conducted using Tensorflow [62] and Keras [63] on an Intel i9900k platform with 128GB RAM and an Nvidia RTX 3090.

The topology of the federation plays a fundamental role within the application of the proposed algorithm, regardless of the particular m-hop protocol taken into account. Randomly generated connected federations constituted by five to twenty clients were considered for our testing, as depicted in Fig. 3.

We remark that the use of multi-hop communications within FedLCon can be applied without loss of generality to both directed and undirected graphs with an arbitrary

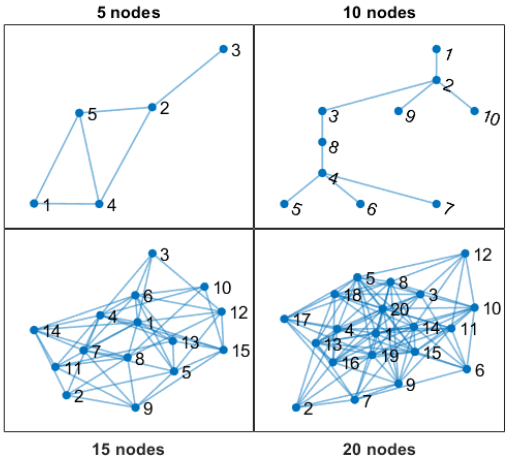

FIGURE 3. Considered undirected graphs: Top left - 5 nodes; Top right - 10 nodes; Bottom left - 15 nodes; Bottom right - 20 nodes.

TABLE 1. Number of consensus iteration n_ϵ evaluated via (10) per communication protocol for the considered scenarios. The standard implementation of FedLCon [6], that is equivalent to a 1-hop implementation of the proposed multi-hop FedLCon algorithm, requires three times more consensus rounds compared to the 2-hop version on all considered federations.

	n_ϵ	
# nodes	FEDLCON [6]	**2-hop FEDLCON**
5	15	**5**
10	30	**10**
15	45	**15**
20	60	**20**

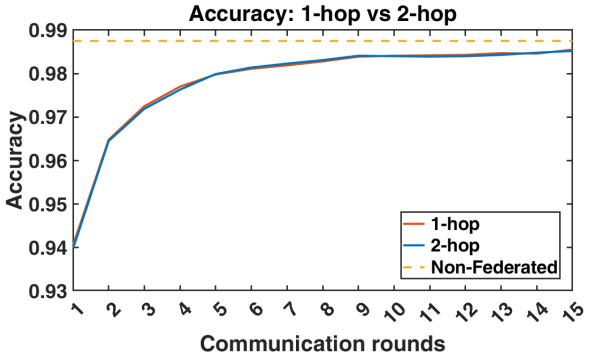

FIGURE 4. MNIST: 10 clients. Accuracy comparison of FedLCon in *1-hop* vs *2-hop*.

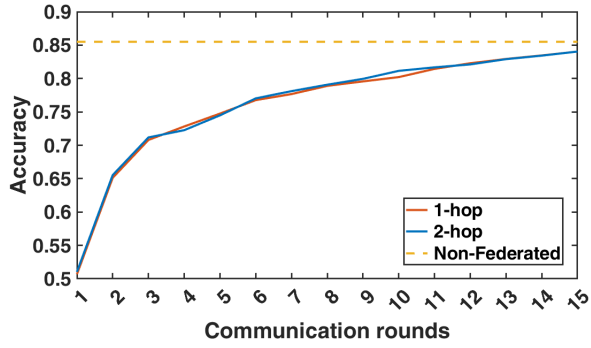

FIGURE 5. BloodMNIST: 20 clients. Accuracy comparison of FedLCon in *1-hop* vs *2-hop*.

number of clients by means of an appropriate choice of the matrix H_p.

For the sake of comparison, we report in Table 1 the evaluation of the number of consensus iterations necessary to assure convergence, that is n_ϵ as defined in (10). From the table, it can be seen that the proposed 2-hop version of FedLCon reduces, for every federation topology considered, the communication overhead by 2/3. We mention that, in larger federations in which n_ϵ may increase significantly, m-hop extensions of (13), such as a 3-hop consensus law, may be used to further reduce n_ϵ.

Looking at (10), this follows from the calculation of matrix H_p; while in the 1-hop case the Laplacian L which is set to be used is the of the undirected graph G, in the 2-hop one is that of the joint undirected graph $\tilde{G}$. The main implication of this effect is that, given the same scenario and computing power, the 2-hop protocol requires less time and communications to complete a consensus round with respect to the 1-hop case.

B. RESULTS

In this subsection we propose a comparison of FedLCon-based on 1-hop and 2-hop communication protocols performances in two specific scenarios, respectively with $|I| = 10$ for MNIST as shown in Fig. 4, and $|I| = 20$ for BloodMNIST in Fig. 5.

Given that a classification problem has to be solved for both applications, i.e., an image is associated with a class to which it belongs, the main metric used for comparative purposes is the accuracy, namely the percentage of correct classifications.

The figures show that, regardless of the particular communication protocol considered, 1-hop (orange line) and 2-hop (blue line), in both cases there is a continuous increase in accuracy passing from one communication round to the next, for all 15 rounds. We remark that both algorithms almost exactly reconstruct the same performance that a centralized architecture using FedAvg would have attained. In the figures we also report with a dashed line the performance that a non-federated system, trained on the entirety of the data, was able to archive after an equivalent number of total training epochs.

We further stress that the almost identical curves between the 1 and 2-hop implementation can be attained thanks to the decoupling of the training and consensus processes allowed by FedLCon. In fact, the consensus value of each consesus round is independent of the particular communication protocol considered, as in both cases it depends only on the initial conditions of the clients' state. This allows to assure that, if provided with the same data, the various clients will share the exact same DeepNN at the end of every consensus step, with no significant degradation of performance.

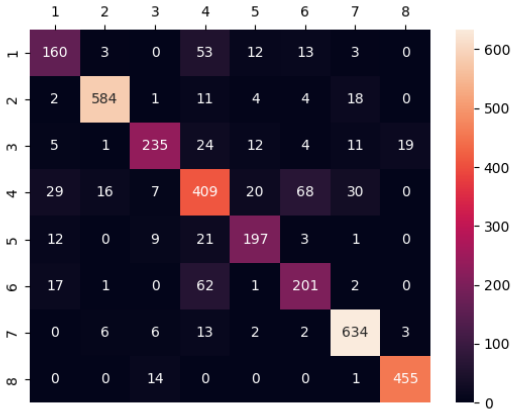

FIGURE 6. **BloodMNIST: Confusion matrix. 20 clients, *2-hop* case.**

The slight variations in accuracy that can be noted sporadically (e.g., rounds 4, 7 and 10 in Fig. 5), can be explained by the 99% convergence assured by the n_ϵ consensus round. This limited convergence error leads to a negligible maximum performance difference that was observed to be at most 1.17 %.

For compliteness, being the BloodMNIST dataset imbalanced, we report in Figure 6 the full confusion matrix obtained on it by the final DeepNN on the federation formed by 20 clients.

VI. CONCLUSION AND FUTURE WORKS

This paper has presented a multi-hop consensus based protocol for decentralized Federated Learning (FL). The proposed approach has been designed to allow for the removal of the centralized server that is required by all standard, centralized, FL solutions that implement a model-averaging procedure. The resulting decentralized federation is guaranteed to reach exactly the same performance of the starting centralized solution, thanks to an almost exact reconstruction of the averaged model, at a faster convergence rate. The properties of the algorithm, including the performance guarantee, have been proven with standard Lyapunov arguments from discrete-time dynamical system theory. Compared to existing decentralized algorithms, the multi-hop consensus protocol implemented in this study significantly reduces the number of required information exchanges, allowing for the algorithm deployment on larger federations and more time-critical applications. Two test cases were discussed to empirically validate the characteristics and capabilities of the proposed algorithm, with an observed performance drop in the order of 1% compared to the centralized case. Future developments involve the design of a decentralized consensus protocol suitable for delay-critical applications and massive scenarios such as large-scale IoT networks.

CONFLICT OF INTEREST

The authors declare no conflict of interest.

REFERENCES

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

[2] A. Blanco-Justicia, J. Domingo-Ferrer, S. Martínez, D. Sánchez, A. Flanagan, and K. E. Tan, "Achieving security and privacy in federated learning systems: Survey, research challenges and future directions," *Eng. Appl. Artif. Intell.*, vol. 106, Nov. 2021, Art. no. 104468.

[3] J. Liu, X. Liang, R. Yang, Y. Luo, H. Lu, L. Li, S. Zhang, and S. Yang, "Federated learning-based vertebral body segmentation," *Eng. Appl. Artif. Intell.*, vol. 116, Nov. 2022, Art. no. 105451.

[4] 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.

[5] Z. Chen, D. Li, J. Zhu, and S. Zhang, "DACFL: Dynamic average consensus based federated learning in decentralized topology," 2021, *arXiv:2111.05505*.

[6] A. Giuseppi, S. Manfredi, and A. Pietrabissa, "A weighted average consensus approach for decentralized federated learning," *Mach. Intell. Res.*, vol. 19, no. 4, pp. 319–330, Aug. 2022.

[7] Z. Jin and R. M. Murray, "Multi-hop relay protocols for fast consensus seeking," in *Proc. 45th IEEE Conf. Decis. Control*, Dec. 2006, pp. 1001–1006.

[8] 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.

[9] N. A. Lynch, *Distributed Algorithms*. Amsterdam, The Netherlands: Elsevier, 1996.

[10] M. H. Degroot, "Reaching a consensus," *J. Amer. Stat. Assoc.*, vol. 69, no. 345, pp. 118–121, Mar. 1974.

[11] J. A. Benediktsson and P. H. Swain, "Consensus theoretic classification methods," *IEEE Trans. Syst., Man, Cybern.*, vol. 22, no. 4, pp. 688–704, Jul./Aug. 1992.

[12] S. C. Weller and N. C. Mann, "Assessing rater performance without a 'gold standard' using consensus theory," *Med. Decis. Making*, vol. 17, no. 1, pp. 71–79, Feb. 1997.

[13] V. Borkar and P. Varaiya, "Asymptotic agreement in distributed estimation," *IEEE Trans. Autom. Control*, vol. AC-27, no. 3, pp. 650–655, Jun. 1982.

[14] J. N. Tsitsiklis, "Problems in decentralized decision making and computation," Massachusetts Inst. Technol., Cambridge Lab Inf. Decis. Syst., Cambridge, MA, USA, 1984.

[15] J. Tsitsiklis, D. Bertsekas, and M. Athans, "Distributed asynchronous deterministic and stochastic gradient optimization algorithms," *IEEE Trans. Autom. Control*, vol. AC-31, no. 9, pp. 803–812, Sep. 1986.

[16] D. P. Bertsekas and J. N. Tsitsiklis, *Parallel and Distributed Computation: Numerical Methods*. Upper Saddle River, NJ, USA: Prentice-Hall, 2003.

[17] C. Wu, F. Wu, L. Lyu, Y. Huang, and X. Xie, "Communication-efficient federated learning via knowledge distillation," *Nature Commun.*, vol. 13, no. 1, Apr. 2022, Art. no. 2032.

[18] H.-P. Wang, S. Stich, Y. He, and M. Fritz, "ProgFed: Effective, communication, and computation efficient federated learning by progressive training," in *Proc. 39th Int. Conf. Mach. Learn.*, vol. 162, K. Chaudhuri, S. Jegelka, L. Song, C. Szepesvari, G. Niu, and S. Sabato, Eds., Jul. 2022, pp. 23034–23054.

[19] Y. Xiong, R. Wang, M. Cheng, F. Yu, and C.-J. Hsieh, "FedDM: Iterative distribution matching for communication-efficient federated learning," Univ. California, Los Angeles, Los Angeles, CA, USA, Tech. Rep., 2022.

[20] C. Li, X. Zeng, M. Zhang, and Z. Cao, "PyramidFL: A fine-grained client selection framework for efficient federated learning," in *Proc. 28th Annu. Int. Conf. Mobile Comput. Netw.*, Oct. 2022, pp. 158–171.

[21] X. Xu, W. Liu, Y. Zhang, X. Zhang, W. Dou, L. Qi, and M. Z. A. Bhuiyan, "PSDF: Privacy-aware IoV service deployment with federated learning in cloud-edge computing," *ACM Trans. Intell. Syst. Technol.*, vol. 13, no. 5, pp. 1–22, Oct. 2022.

[22] Y. M. Saputra, D. Nguyen, H. T. Dinh, Q.-V. Pham, E. Dutkiewicz, and W.-J. Hwang, "Federated learning framework with straggling mitigation and privacy-awareness for AI-based mobile application services," *IEEE Trans. Mobile Comput.*, early access, May 30, 2022, doi: 10.1109/TMC.2022.3178949.

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

[24] X. Yuan and P. Li, "On convergence of FedProx: Local dissimilarity invariant bounds, non-smoothness and beyond," in *Advances in Neural Information Processing Systems*, vol. 35, S. Koyejo, S. Mohamed, A. Agarwal, D. Belgrave, K. Cho, and A. Oh, Eds. Red Hook, NY, USA: Curran Associates, 2022, pp. 10752–10765.

[25] A. T. Nguyen, P. Torr, and S. N. Lim, "FedSR: A simple and effective domain generalization method for federated learning," in *Advances in Neural Information Processing Systems*, vol. 35, S. Koyejo, S. Mohamed, A. Agarwal, D. Belgrave, K. Cho, and A. Oh, Eds. Red Hook, NY, USA: Curran Associates, 2022, pp. 38831–38843.

[26] Y. Zhang, D. Zeng, J. Luo, Z. Xu, and I. King, "A survey of trustworthy federated learning with perspectives on security, robustness, and privacy," 2023, *arXiv:2302.10637*.

[27] J. Zhang, J. Zhou, J. Guo, and X. Sun, "Visual object detection for privacy-preserving federated learning," *IEEE Access*, vol. 11, pp. 33324–33335, 2023.

[28] K. Prokop, D. Połap, G. Srivastava, and J. C.-W. Lin, "Blockchain-based federated learning with checksums to increase security in Internet of Things solutions," *J. Ambient Intell. Hum. Comput.*, vol. 14, no. 5, pp. 4685–4694, Sep. 2022.

[29] S. Gao, J. Luo, J. Zhu, X. Dong, and W. Shi, "VCD-FL: Verifiable, collusion-resistant, and dynamic federated learning," *IEEE Trans. Inf. Forensics Security*, vol. 18, pp. 3760–3773, 2023.

[30] H. Ye, L. Liang, and G. Y. Li, "Decentralized federated learning with unreliable communications," *IEEE J. Sel. Topics Signal Process.*, vol. 16, no. 3, pp. 487–500, Apr. 2022.

[31] F. P. Sánchez, M. R. Pedruelo, C. C. Casamayor, and A. P. Chust, "Convergence of weighted-average consensus for undirected graphs," *J. Complex Syst. Sci.*, vol. 4, no. 1, pp. 13–16, 2014.

[32] J. A. Fax, *Optimal and Cooperative Control of Vehicle Formations*. Pasadena, CA, USA: California Institute of Technology, 2002.

[33] J. A. Fax and R. M. Murray, "Information flow and cooperative control of vehicle formations," *IEEE Trans. Autom. Control*, vol. 49, no. 9, pp. 1465–1476, Sep. 2004.

[34] R. O. Saber and R. M. Murray, "Consensus protocols for networks of dynamic agents," California Inst. Technol., Pasadena, CA, USA, Tech. Rep., 2003.

[35] 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.

[36] J. A. Fax and R. M. Murray, "Graph Laplacians and stabilization of vehicle formations," *IFAC Proc. Volumes*, vol. 35, no. 1, pp. 55–60, 2002.

[37] R. Olfati-Saber and R. M. Murray, "Distributed cooperative control of multiple vehicle formations using structural potential functions," *IFAC Proc. Volumes*, vol. 35, no. 1, pp. 495–500, 2002.

[38] N. E. Leonard and E. Fiorelli, "Virtual leaders, artificial potentials and coordinated control of groups," in *Proc. 40th IEEE Conf. Decis. Control*, vol. 3, Dec. 2001, pp. 2968–2973.

[39] K. D. Listmann, M. V. Masalawala, and J. Adamy, "Consensus for formation control of nonholonomic mobile robots," in *Proc. IEEE Int. Conf. Robot. Autom.*, May 2009, pp. 3886–3891.

[40] L.-Y. Lu and C.-C. Chu, "Consensus-based droop control synthesis for multiple DICs in isolated micro-grids," *IEEE Trans. Power Syst.*, vol. 30, no. 5, pp. 2243–2256, Sep. 2015.

[41] J. Khazaei and Z. Miao, "Consensus control for energy storage systems," *IEEE Trans. Smart Grid*, vol. 9, no. 4, pp. 3009–3017, Jul. 2018.

[42] A. Goldsmith, *Wireless Communications*. Cambridge, U.K.: Cambridge Univ. Press, 2005.

[43] Z. Zhao and Z. Lin, "Global leader-following consensus of a group of general linear systems using bounded controls," *Automatica*, vol. 68, pp. 294–304, Jun. 2016.

[44] S. Manfredi, "Design of a multi-hop dynamic consensus algorithm over wireless sensor networks," *Control Eng. Pract.*, vol. 21, no. 4, pp. 381–394, Apr. 2013.

[45] M. Zhu and S. Martínez, "Discrete-time dynamic average consensus," *Automatica*, vol. 46, no. 2, pp. 322–329, Feb. 2010.

[46] B. Wang, J. Fang, H. Li, X. Yuan, and Q. Ling, "Confederated learning: Federated learning with decentralized edge servers," *IEEE Trans. Signal Process.*, vol. 71, pp. 248–263, 2023.

[47] L. Barbieri, O. Simeone, and M. Nicoli, "Channel-driven decentralized Bayesian federated learning for trustworthy decision making in D2D networks," in *Proc. IEEE Int. Conf. Acoust., Speech Signal Process. (ICASSP)*, Jun. 2023, pp. 1–5.

[48] Y. Shi, L. Shen, K. Wei, Y. Sun, B. Yuan, X. Wang, and D. Tao, "Improving the model consistency of decentralized federated learning," 2023, *arXiv:2302.04083*.

[49] M. Costantini, G. Neglia, and T. Spyropoulos, "FedDec: Peer-to-peer aided federated learning," 2023, *arXiv:2306.06715*.

[50] F. Chen, G. Long, Z. Wu, T. Zhou, and J. Jiang, "Personalized federated learning with graph," 2022, *arXiv:2203.00829*.

[51] A. Z. Tan, H. Yu, L. Cui, and Q. Yang, "Towards personalized federated learning," *IEEE Trans. Neural Netw. Learn. Syst.*, early access, Mar. 28, 2022, doi: 10.1109/TNNLS.2022.3160699.

[52] K. Wang, Q. He, F. Chen, C. Chen, F. Huang, H. Jin, and Y. Yang, "FlexiFed: Personalized federated learning for edge clients with heterogeneous model architectures," in *Proc. ACM Web Conf.*, Apr. 2023, pp. 2979–2990.

[53] Q. Li, Z. Wen, Z. Wu, S. Hu, N. Wang, Y. Li, X. Liu, and B. He, "A survey on federated learning systems: Vision, hype and reality for data privacy and protection," *IEEE Trans. Knowl. Data Eng.*, vol. 35, no. 4, pp. 3347–3366, Apr. 2023.

[54] T. Li, A. K. Sahu, M. Zaheer, M. Sanjabi, A. Talwalkar, and V. Smith, "Federated optimization in heterogeneous networks," in *Proc. Mach. Learn. Syst.*, vol. 2, 2020, pp. 429–450.

[55] D. S. Bernstein, *Matrix Mathematics*. Princeton, NJ, USA: Princeton Univ. Press, 2009.

[56] K. Ogata, *Discrete-Time Control Systems*. Upper Saddle River, NJ, USA: Prentice-Hall, 1995.

[57] S. Manfredi, "A theoretical analysis of multi-hop consensus algorithms for wireless networks: Trade off among reliability, responsiveness and delay tolerance," *Ad Hoc Netw.*, vol. 13, pp. 234–244, Feb. 2014.

[58] 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.

[59] J. Yang, R. Shi, D. Wei, Z. Liu, L. Zhao, B. Ke, H. Pfister, and B. Ni, "MedMNIST v2—A large-scale lightweight benchmark for 2D and 3D biomedical image classification," 2021, *arXiv:2110.14795*.

[60] A. Acevedo, A. Merino, S. Alférez, Á. Molina, L. Boldú, and J. Rodellar, "A dataset of microscopic peripheral blood cell images for development of automatic recognition systems," *Data Brief*, vol. 30, Jun. 2020, Art. no. 105474.

[61] A. F. Agarap, "Deep learning using rectified linear units (ReLU)," 2018, *arXiv:1803.08375*.

[62] M. Abadi et al. (2015). *TensorFlow: Large-Scale Machine Learning on Heterogeneous Systems*. [Online]. Available: https://tensorflow.org

[63] F. Chollet. (2015). *Keras*. [Online]. Available: https://github.com/fchollet/keras

DANILO MENEGATTI received the master's degree in control engineering from the University of Rome "La Sapienza," in 2020, where he is currently pursuing the Ph.D. degree in automatic control, bioengineering and operations research with the Department of Computer, Control, and Management Engineering "Antonio Ruberti" (DIAG). His research interests include intelligent systems, distributed learning, and reinforcement learning applications.

ALESSANDRO GIUSEPPI (Senior Member, IEEE) received the master's degree in control engineering and the Ph.D. degree in automatica from the University of Rome "La Sapienza," in 2016 and 2019, respectively. He is currently an Assistant Professor (RTDa) with the Department of Computer, Control, and Management Engineering "Antonio Ruberti" (DIAG), University of Rome "La Sapienza." Since 2016, he has been participated in six other EU and national research projects, mainly in the fields of control systems and artificial intelligence. Currently, he is the Scientific Coordinator of the ESA-Funded Research Project ARIES, related to wildfire emergency management and the Work Package Leader of the EU-Korea H2020 Project 5G-ALLSTAR. Since 2020, he has been serving as an Associate Editor for the *International Journal of Control, Automation and Systems* (Springer). His research interests include network control and intelligent systems, where he published about 50 papers in international journals and conferences.

SABATO MANFREDI is currently an Associate Professor in automatic control with the Department of Electrical Engineering and Information Technology, University of Naples Federico II, Naples, Italy. He has authored/co-authored more than 100 scientific publications including the monograph: Multilayer Control of Networked Cyber-Physical Systems. Application to Monitoring, Autonomous and Robot Systems (Advances in Industrial Control Series, Springer, 2017). He collaborates with many international universities (including Imperial College, London) and companies, holds European patent and is involved in a range of academic, industrial and consulting projects. His research interests include automatic control with a special emphasis on distributed estimation, control and optimization of/over cyber-physical systems and complex networks, decentralised machine learning techniques for intelligent networked systems and smart cities.

ANTONIO PIETRABISSA (Senior Member, IEEE) received the degree in electronics engineering and the Ph.D. degree in systems engineering from the University of Rome "La Sapienza," in 2000 and 2004, respectively. He is currently an Associate Professor with the Department of Computer, Control, and Management Engineering "Antonio Ruberti" (DIAG), University of Rome "La Sapienza," where he teaches automatic control and process automation. Since 2000, he has been participating in about 25 EU and national research projects. He is also the Coordinator of the Project ARIES on fire emergency prevention, funded by ESA, and the scientific responsible of the research projects 5GALLSTARS on 5G communications, funded within the H2020 Europe-South Korea Cooperation, and FedMedAI on medical applications of federated learning, funded by region Lazio (IT). He is the author of more than 50 journal articles and 80 conference papers. His research interests include the application of systems and control theory to the analysis and control of networks. He serves as an Associate Editor for *Control Engineering Practice* (Elsevier) and IEEE TRANSACTIONS ON AUTOMATION SCIENCE AND ENGINEERING.

Open Access funding provided by 'Università degli Studi di Roma "La Sapienza" 2' within the CRUI CARE Agreement