Mitigating the Communication Straggler Effect in Federated Learning via Named Data Networking

Marica Amadeo, Claudia Campolo, Antonella Molinaro, Giuseppe Ruggeri, and Gurtaj Singh

The authors address issues in federated learning related to devices experiencing poor channel conditions.

Abstract

Federated learning (FL) has emerged as a prominent solution that enables distributed training of machine learning (ML) models at multiple end-devices with their own data samples. The FL performance, and in particular, the training convergence speed, are limited by the device with the lowest computation and communication capabilities typically referred to as a straggler. In this work, we address the issues related to the presence of communication stragglers, that is, devices experiencing poor channel conditions. By leveraging named data networking (NDN) and customizing its forwarding fabric to improve the ML model delivery in the presence of lossy communications, our solution allows potential stragglers to beneficially participate in the FL application. Results show the advantages of the conceived solution in terms of reduced training time when compared to a conventional host-centric FL approach, and higher accuracy with reference to the case in which stragglers are not selected.

Introduction

Federated Learning (FL) has been proposed by Google to enable distributed Machine Learning (ML) model execution [1]. According to this paradigm, distributed clients (typically mobile devices), coordinated by an aggregator server, collaboratively train a shared ML model by using their own private data. The training results, instead of raw datasets, are sent to the aggregator server to update the global model. The same process continues iteratively for many rounds, until the requested accuracy is reached.

Despite the well-known advantages, FL faces *communication* challenges. In particular, some clients may experience connectivity issues due to unreliable and lossy links, thus becoming *communication stragglers* that deteriorate the FL convergence [1]. To mitigate such effect, a variety of client selection mechanisms have been proposed that exclude stragglers from training [2]. This however results in a decrease in the amount of exploited training data and a consequent slow down of the FL convergence. An alternative solution to avoid learning performance loss is not to exclude the communication stragglers but to improve their performance [3]. In this context, some physical or medium access control (MAC) layer strategies optimize data transmissions and recover fast from losses caused by the presence of stragglers [1]. Other recent works investigated synergies with technologies like collaborative relays [3], Reconfigurable Intelligent Surfaces (RISs) [4], and Unmanned Aerial Vehicles (UAVs) [5].

In this article, we discuss a new approach *to enhance the FL performance* and *cope with the slow convergence caused by communication stragglers*. Our solution is based on the Named Data Networking (NDN) future Internet paradigm [6], which provides name-based data dissemination through the exchange of *Interest* and *Data* packets and natively supports multicast data delivery and in-network caching.

In [7] NDN is argued as a key enabler for in-network intelligence; requests are expressed as unique names conveyed in Interest packets and are satisfied by intermediate nodes. In our previous work in [8], the strengths of NDN in supporting FL procedures were theoretically discussed and a framework, named *NDN-FL*, was conceived to enhance NDN routines and enable FL client discovery and selection. Moreover, there, early encouraging results showed the improvements achieved by the vanilla NDN procedures against a conventional application-layer approach, under the simplified assumption of homogeneous clients.

This article provides the following main original contributions:

- We design *enhanced NDN-FL*, *eNDN-FL*, which builds upon our previous work in[8] for what concerns client discovery and selection; and goes beyond that by customizing the NDN forwarding and caching routines to make the model and its parameters exchange between FL clients and aggregator more robust and quicker. Heterogeneous clients are explicitly considered, with some of them experiencing poor connectivity and potentially acting as communication stragglers. To this aim, nodes located nearby the straggler (both intermediate nodes in the path toward the aggregator as well as neighboring clients) may serve it with a cached copy of the global model and/or its updates.

ISSN: 0163-6804
Digital Object Identifier: 10.1109/MCOM.001.2300419

Marica Amadeo, Claudia Campolo, Giuseppe Ruggeri, and Gurtaj Singh are with the University of Reggio Calabria, Italy and also with CNIT, Italy; Antonella Molinaro is with the University Mediterranea of Reggio Calabria, Italy, CNIT, Italy, and Université Paris-Saclay, France.

Moreover, caching FL data at intermediate nodes is judiciously adapted to the dynamics of the training rounds, not to waste precious storage resources.
- We assess the performance of the proposal, through a validation study conducted with the ndnSIM simulator and realistic datasets, and analyze the results in terms of training round latency and accuracy.

Federated Learning

Basics

The vanilla FL protocol follows a synchronous process based on the following steps in each training round, re-iterated until the global training is complete [1].

Initialization: When a training task has to be performed, the server prepares the initial global model and training parameters, such as the learning rate and the number of training iterations. Then, it selects *K* clients, randomly or according to a specific policy, for example, based on the available computation capabilities and/or network conditions [8], or even the quality of the owned dataset [9]. Indeed, the clients' training data are usually non-independently and identically distributed (non-IID), and imbalanced, that is, some clients can have large datasets, while others can have only a few records.

Model Dissemination and Local Training: The server disseminates the global model and settings to the selected *K* clients, which train the model on their local dataset and, after completion, push the model update to the server.

Aggregation: After receiving the updates from all *K* clients, the server performs the model aggregation and possibly instructs the next round of training by sending the updated global model to the selected clients.

The Straggler Issue

In a realistic environment, the communication channel is not perfect and clients may experience poor and heterogeneous connectivity conditions [1]. Synchronous FL can be significantly affected by communication stragglers, since the slowest client dictates the training pace: the server must wait until receiving the training updates from all participants.

Current FL designs typically leverage the reliable Transport Control Protocol (TCP), which re-transmits all the incorrectly received packets [10], thus preserving the model accuracy but causing extra delays. Normally, synchronous FL relies on a maximum training round time, T_{max}: if the model update from a client is not received by the aggregator before the T_{max} deadline, then the contribution from that client is excluded from the training round. To understand the TCP performance under diverse network and packet loss conditions, we considered a simple scenario with clients directly connected to the aggregator through the 3G, 4G or WiFi technologies. For each of them, we assumed bandwidth limits set, respectively, equal to 5, 20 and 60 Mb/s [10], and packet losses from 0.2% to 10%.

Figure 1 shows, for each technology, the average training round duration achieved by a TCP-based FL, for different packet loss values. It can be

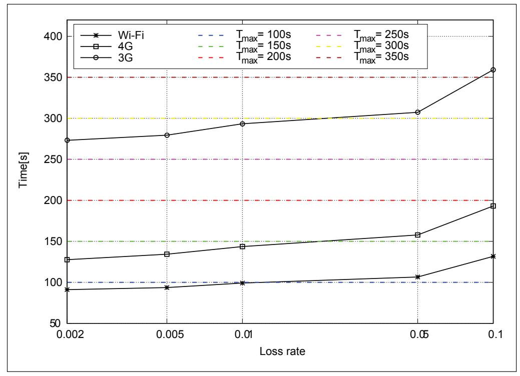

FIGURE 1. Training round time vs. packet loss rate for different communication technologies.

Current FL designs typically leverage the reliable Transport Control Protocol (TCP), which re-transmits all the incorrectly received packets [10], thus preserving the model accuracy but causing extra delays.

observed that the training round time increases with the packet losses, whatever the considered technology. Different T_{max} values, in the range between 100 and 350 s, are reported in the figure. For packet losses below 1% and Tmax values below 300 s, the average training round duration, for all technologies, is within the deadlines. This means that all clients would be able to download the model, train it locally and report the update before being excluded by the aggregator. This also implies that, within the mentioned loss and T_{max} ranges, accuracy remains unaffected. Out of these ranges, accuracy will gradually decrease because some clients will no longer be able to complete the training round within T_{max}.

Related Work

To cope against the straggler effect, previous efforts in the literature proposed *asynchronous* FL schemes, which however may limit the model utility or also diverge the training process, due to the potential staleness of client's updates, unless properly handled [11].

Compression techniques encompassing, among others, quantization, sparsification, knowledge distillation, may reduce the amount of transmitted bits, by addressing the communication issues [1, 12]. However, such solutions require substantial changes at the application, since they are directly implemented at the algorithmic level of FL and may come at the expenses of a loss of accuracy [10].

With similar objective of reducing the pressure on the network, semantic communications can deliver data more efficiently over wireless links by differentiating the usefulness of training data samples from different clients [13]. Such solutions may require computationally intensive ML models, in their turn, to understand which information is of interest before transmitting it.

The proposals in [3] and references therein introduce collaborative relays, which receive updates from nearby stragglers and forward them to the aggregator. However, these works do not present a specific communication protocol supporting the nodes' relaying capability. In addition, the focus is on model updates delivery (i.e., uplink transmissions), while communication issues during the global model distribution (i.e., downlink transmissions) are not considered.

As a result, FL still needs a robust and effective communication architecture that improves

As a result, the node aggregates the same requests in the PIT and forwards a single one over the same link, thus reducing the load in the network and on the server.

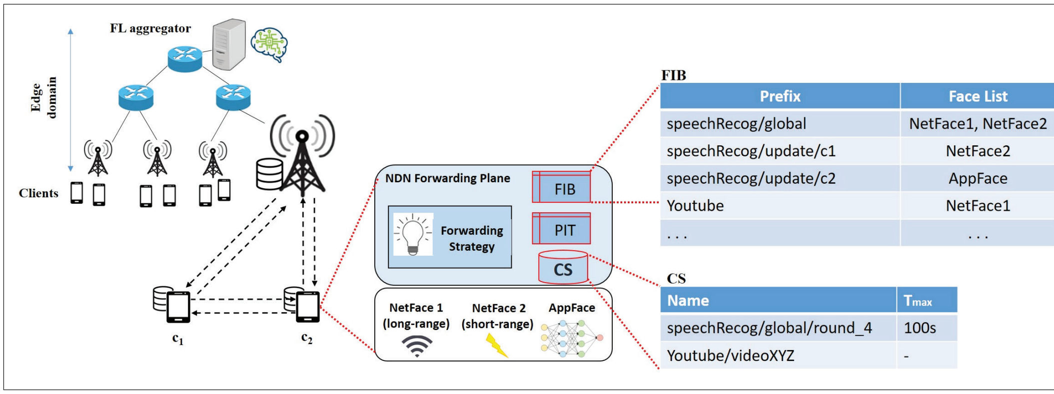

FIGURE 2. Reference scenario and NDN forwarding plane at client c_2.

client-server content distribution. In the following, we introduce how the NDN paradigm may address the mentioned issues.

Named Data Networking

Basics

NDN is an information-centric communication architecture based on the exchange of two named packets: the Interest, sent by consumers to request contents, and the Data, sent by any content provider, that is, the original source or a cacher, to answer the request [6]. Interests carry application-level Uniform Resource Identifier (URI)-like content names that are directly used by network routers to forward the request toward the source of the corresponding Data.

Each NDN node *n* maintains three tables:

- A Forwarding Information Base (FIB) that lists, per each name prefix, the outgoing interface(s) and the corresponding performance metric(s), for example, the round-trip-time (RTT)
- A Pending Interest Table (PIT), listing the forwarded Interests that are waiting to be consumed by a Data packet
- A Content Store (CS) to cache incoming Data packets.

Upon receiving an Interest, *n* first looks in the CS for a matching packet to immediately serve the consumer. If the CS matching fails, *n* looks in the PIT. If a matching is found, *n* discards the Interest, since an equal request has been already transmitted and it is waiting to be consumed by the data. As a result, the node aggregates the same requests in the PIT and forwards a single one over the same link, thus reducing the load in the network and on the server. If the PIT matching fails, *n* creates a new entry in the PIT and looks in the FIB to forward the Interest.

Stateful Forwarding and *on-path* caching are two notable and highly customizable NDN features, directly provided at the *network layer*, that can largely improve communication performance.

Stateful Forwarding

The PIT allows filtering unsolicited Data packets, that is, Data without a matching PIT entry are discarded, and aggregating Interests for the same content, thus enforcing multicast delivery. This largely *reduces the traffic congestion and the load on the original source*. In parallel, stateful forwarding allows *fast recovery* from *losses*. If a pending Interest is not consumed by the Data within the expected time interval related to an outgoing face, the node can try alternative request paths. The request is finally discarded when the pre-defined time-to-live of the Interest (TTL_{Int}) expires.

Caching

In-network caching can be crucial to *speed up content delivery*, especially in challenging communication scenarios, for example, in case of packet losses. Unlike existing systems based on the TCP/IP protocols, where caching is supported in specific nodes at the *application layer*, NDN data can be re-transmitted by any *on-path cacher*, that is, any node in the delivery path between consumers and sources. Several caching policies can be defined according to the reference application and network environment. NDN caching can also deal with transient contents, that is, data that, after a certain time period, become invalid.

The NDN source can specify the TTL of its contents and include it in the *FreshnessPeriod* field of the Data packets' header. When the TTL expires, the CS removes the cached packets.

Our Proposal

Main Assumptions

Reference Scenario: A hierarchical edge network topology is considered, where leaf nodes act as Access Points (APs) for a set of mobile devices which may act as clients for the sake of a model training task (Fig. 2). The server is co-located with the edge root node and may reach the clients via multi-hop paths. The approach can be however applied to any edge network topology.

We assume that synchronous FL is implemented, with training round deadline set to T_{max} [2].

Potential clients are equipped with multiple network access interfaces, for example, fourth generation (4G) / fifth generation (5G) cellular, WiFi, to connect to the closer AP or to other neighboring clients. In the following, we will refer to the interface toward the AP as *long-range*, and to the one toward a nearby client as *short-range*.

NDN Architecture: All the nodes in the reference scenario implement NDN and use Interest and Data packets for the exchange of the global model and local model updates. NDN can be implemented as a clean-slate solution if the edge domain is a greenfield deployment, where new networking solutions can be easily implemented from scratch over the access layer technologies. Alternatively, thanks to softwarized network designs, which facilitate the implementation of multiple protocol stacks over the same device, the NDN paradigm could co-exist with other networking solutions, for example, IP-based, thus avoiding any additional hardware cost.

A legacy NDN routing protocol is run by each node to discover and maintain adjacency relations with the neighbors, build paths toward model sources and detect link failures. Depending on the phase of the FL process, clients may act as NDN data consumers or sources: they are consumers during the global model retrieval, and they behave as sources when transmitting the model updates to the server. Similarly, the server acts as a source of the global model and as a consumer of model updates.

Contributions: *eNDN-FL* boosts FL data dissemination by enabling multi-path forwarding and off-path caching. Clients can retrieve the global model in a multi-path fashion, that is, from the server; from an intermediate node acting as *on-path cacher*; or from a neighboring client acting as off-path cacher. The latter option is enabled by *eNDN-FL* by leveraging the existing NDN routing advertisement messages, without introducing additional signalling overhead. This is an important departure from conventional TCP-like approaches, where FL clients have to establish an end-to-end session with the server in order to get the global model. In addition, *eNDN-FL* defines a *round-limited cache reservation* mechanism that promptly releases the storage resources used for the global model as soon as it becomes obsolete, that is, after a given training round deadline. *eNDN-FL* also supports model updates retrieval from the clients via multiple uplink paths, that is from the client through the AP when the link quality is good; or through a neighboring client acting as a *relay* over a higher-quality link.

There is a twofold reward for clients acting as cachers and relays. By contributing to make the overall FL convergence faster: they get an updated better trained version of the model in a shorter time; and they reduce the number of training rounds, by saving their computing resources at the expenses of the forwarding operation.

The conceived extensions are detailed in the following.

Initialization

Before starting the FL task, the server has to discover and select the set of clients to involve in. To this aim, it follows the procedure devised in our previous work [8], which is based on the exchange of special Interest and Data packets, respectively advertising the FL application and its main parameters (e.g., task type, required accuracy) and the potential clients' capabilities. In this work, we consider that the set of clients has been already discovered and selected by the FL server as in [8], and we focus on the next steps for model sharing and updating, and on naming convention.

eNDN-FL exploits the NDN hierarchical naming scheme to identify the global model and its updates. The following naming convention is used for the global model: *FL_app/global/round_i*, where the name prefix *FL_app* identifies the type of FL application running at the server (e.g., speech recognition, object detection), the label *global* refers to the aggregated model generated by the FL server, and the last name component *round_i* identifies the specific training round i. As an example, the name *speechRecog/global/round_1* identifies the global model at the first round of an FL speech recognition application. Conversely, the naming scheme of the model updates generated by a given client k consists of the same name prefix, namely *FL_app*, used by the server, followed by the label *update* that refers to the updated model trained locally, and by the identifiers of the client producing the content and the corresponding training round, that is, *FL_app/update/client_k/round_i*.

FIB Configuration and Forwarding

Once the clients join the FL application, they add a new FIB entry coupling the global model name prefix with the default outgoing interface to reach the server, for example, through the AP. Then, they start periodically advertising themselves as (future) producers of model updates by sending routing messages over all the available short-/long-range interfaces. In *eNDN-FL*, the advertisements of a client c_1 have a twofold target:

- Allowing the other receiving edge nodes (including the server and neighboring clients) to identify a source of model updates and, therefore, to fill their FIB accordingly
- Making the other clients aware of its availability to act as: *off-path cacher* of the global model; or *relay* for model updates from neighboring clients (potential communication stragglers).

Therefore, when receiving the advertisement from a client c_1, the forwarding strategy of a neighboring client c_2 will include in the FIB a new entry for reaching the model updates, and modify the existing entry for the global model, by adding the short-range outgoing interface toward client c_1. Entries are removed if advertisements are no longer received, for example, due to mobility. In terms of overhead in the FIB, compared to the traditional NDN implementation, *eNDN-FL* introduces additional outgoing interfaces information, that is, a few bytes, per each FIB entry. Therefore, the required memory cost is contained.

As an example, in Fig. 2, c_2 has three entries in its FIB related to the running FL application. The first one, <*speechRecog/global*, *NetFace1*, *NetFace2*>, records the main prefix of the global model and the corresponding outgoing network interfaces over which sending Interests to retrieve it. In addition to the long-range connectivity toward the AP, c_2 has recorded the short-range connectivity toward c_1, which participates in the same FL application and therefore, it may become also a provider of the global model. There are not a priori guarantees that c_1 will be able to return the global model, but it could be queried by c_2 in case of communication issues over the direct connectivity link toward the AP. The second entry, <*speechRecog/update/c1*, *NetFace2*>, records the main prefix of the updates provided by c_1 and the corresponding outgoing network interface. This also implies that the AP has two options for retrieving the updates from c_1: through the direct link, or through c_2 acting as relay. Finally, the third entry, <*speechRecog/update/c2*, *AppFace*>, simply indicates that c_2 is the original source of content *speechRecog/update/c2*.

A legacy NDN routing protocol is run by each node to discover and maintain adjacency relations with the neighbors, build paths toward model sources and detect link failures. Depending on the phase of the FL process, clients may act as NDN data consumers or sources: they are consumers during the global model retrieval, and they behave as sources when transmitting the model updates to the server. Similarly, the server acts as a source of the global model and as a consumer of model updates.

When the Data packet is not returned within the expected interface RTT, the AP retransmits the unsatisfied Interest over the alternative interfaces, if available.

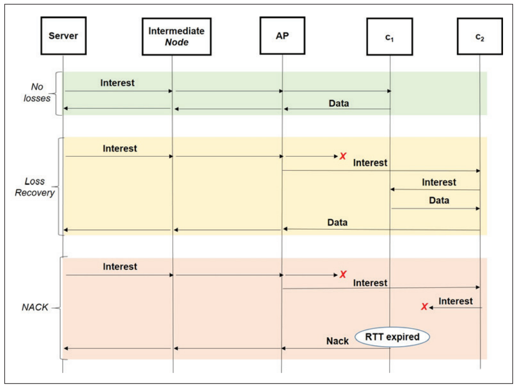

FIGURE 3. Model update retrieval.

Global Model Retrieval

At each training round *i*, clients request the global model by sending Interest packets with name *FL_app/global/round_i* toward the server. Intermediate nodes can aggregate in the PIT the same Interests and transmit in multicast the Data to distinct clients, as natively supported by NDN. For instance, the AP in Fig. 2 could aggregate the same requests for the global model from the two clients at a given training round *i* and then serve them with the same Data packets.

However, due to adverse channel conditions, the direct communication link between the AP and a client could be unavailable. When a client detects this issue, it requests the missing Data to other neighboring clients, according to the FIB configuration. To improve the caching performance and cope against the straggler issue, we introduce a *round-limited cache reservation mechanism*. Each client participating in the FL application must reserve the storage space for caching the global model. At each round the model is cached for a time equal to the training round deadline, T_{max}.

As an example, Fig. 2 shows that c_2 maintains in its CS the global model of the current round with a certain T_{max} duration. If c_1 is experiencing some communication issues with the AP, it can retrieve the global model from c_2.

The NDN Forwarding Strategy module at each client is in charge of handling the Interest retransmissions to retrieve the global model from other nearby clients. In our design, we introduce a *client-aided forwarding policy*, according to which a straggler selects the best nearby client, that is, the one with the lowest RTT, to retrieve the model. If no Data is returned within the expected RTT, the straggler retransmits the Interest over the second best client, if any, and so on. It gives up if the forwarding options run out.

Model Updates Retrieval

To retrieve the model updates at each training round *i*, the server transmits Interest packets toward the selected clients according to the defined naming scheme. Each Interest carries a TTL_{Int} equal to the maximum tolerable time that the server may wait before aggregating the model updates. By doing so, possible late data transmissions are treated as unsolicited packets and are discarded by intermediate nodes, without unnecessarily overloading the server. Indeed, in agreement with the assumed synchronous FL approach, they would have been deleted at the server in any case.

In parallel, to quickly recover from possible data losses and further limit the load on the server, the forwarding strategy of APs and (intermediate) edge nodes is configured to enforce *autonomous retransmissions* of unsatisfied pending Interests with *Negative Acknowledgment (NACK) feedback*. When the Data packet is not returned within the expected interface RTT, the AP retransmits the unsatisfied Interest over the alternative interfaces, if available. If no Data is received, the AP sends back a NACK to announce that the Data cannot be retrieved. At the NACK reception, the previous edge node can further try alternative routes (if available) toward the client. The same process is repeated until either all the available routes are tested or the Interest TTL expires. Figure 3 shows an example of model update retrieval in the presence of: no-loss connectivity toward c_1; data recovery through c_2; failed loss recovery and NACK transmission.

Performance Evaluation

Simulation Tools and Settings

We consider an edge-based FL approach, where a set of clients connect to the edge server, acting as global model aggregator, through an edge network domain, Fig. 2. This latter is modelled as a three-layer infrastructured hierarchical tree topology where the root node is connected to the aggregator. Wired links have latency in the range [1–3 ms]. Clients are wirelessly connected to the leaf nodes, acting as APs, and may experience adverse channel conditions thus becoming *communication stragglers.* In particular, we simulate the presence of 10 clients and randomly select some of them to act as communication stragglers. They experience a variable packet loss rate over the long-range 4G interface toward the AP, and may establish short-range connectivity with nearby clients acting as relays. All clients are assumed with homogeneous computing capabilities. We compare *eNDN-FL* against:

- The *legacy FL* approach, where clients establish TCP-based end-to-end connections with the server
- The *relaying* approach, where legacy FL is augmented with relaying in the uplink path, like in [3],
- The *NDN* baseline approach, where all the nodes implement the NDN architecture with in-network caching and the default best-route forwarding strategy.

To realistically simulate the mentioned approaches in the considered topology, we leverage ndnSIM, a module deployed within ns-3 by the NDN research community. Results are reported with 95% confidence intervals computed over 20 runs.

Dataset and Training Model

We focus on an object recognition learning task when considering the CIFAR-10 dataset [14], which includes 60,000 color images, tagged with one of 10 available, mutually exclusive classes, for example, airplane, automobile. The total size of the dataset is 57 MB. The used neural network (NN) model is ResNet-50 [15] and its size is 98 MB. The learning rate has been set to 0.22, the number of local epochs to 5 and the mini-batch size to 30.

Two different data distributions are considered for training the NN. For the IID case, the starting dataset is divided equally among the clients, that is, the number of samples per class is identical on each client. For the non-IID distribution, the samples belonging to each class are not distributed equally on all clients; this leads to an imbalance. To avoid accuracy issues in the non-IID case, it is assumed that each client has a minimum number of samples for each class of the starting dataset.

Results

Figure 4 reports the accuracy obtained by setting the maximum number of rounds to 200, for the two different dataset distributions, regardless of the communication approach. We consider two distinct scenarios: the case where all 10 clients contribute to the training procedure (solid lines); and the case where 2 or 5 stragglers are excluded by the training procedure (dashed lines). In the latter case, only 8 or 5 out of 10 clients, respectively, participate to the training; the stragglers are excluded due to communication issues.

As expected, a higher accuracy is achieved for the IID dataset distribution because individual clients have balanced subsets which allow the NN to train uniformly. In the non-IID case, instead, the subsets are unbalanced in terms of the number of samples associated with each class. Hence, the NN trains worse on classes that have fewer samples than others. Furthermore, when the stragglers are not involved in the training procedures, the accuracy performance gets worse. This is especially true for non-IID dataset distribution and for a higher number of stragglers. This confirms that improving communication conditions of all the clients is imperative to make the best of the datasets available at their premises and target high FL training accuracy.

To assess the benefits introduced by eNDN-FL over the benchmark schemes, results in Fig. 5 show the average duration of the training round under different loss conditions perceived by the communication stragglers. The metric is derived as the sum of the time taken on average by a client to retrieve the global model from the server, to train it locally and to send the updated model back to the server. We set the training round deadline, T_{max}, equal to 600 s [2]. Hence, the proposal and the considered benchmarks achieve the same accuracy performance because the same population of clients contribute to the aggregation. The training measurements were performed on a machine equipped with an Intel Core i7-9750H CPU, 16 GB DDR4 RAM, and an NVIDIA GeForce GTX 1050 Ti GPU.

Only the non-IID case is considered in this analysis, being the most affected by the presence of communication stragglers. For all the considered networking solutions, we distinguish the average training round time of stragglers and non-stragglers, that is, clients affected by varying packet loss rate and clients that do not experience communication issues, respectively. It can be observed that the metric for stragglers highly increases with the loss rate and the worst performance is obtained by the legacy approach. Indeed, in a traditional networking solution, based on end-to-end communications without in-network caching, clients must download the global model directly from the server and, in case of data losses, retransmissions are enforced at the server-side. By providing in-network caching, instead, NDN outperforms the legacy and relaying approaches since data can be recovered from the closest on-path cacher, that is, the AP. However, stragglers can further reduce the time needed to download the global model by retrieving it from nearby clients. Moreover, APs can retrieve updates from the stragglers by leveraging nearby clients as intermediate forwarders. Interestingly, the time needed by an eNDN-FL straggler client to download the model, train it and update the server approaches the one experienced by a non-straggler client. This trend confirms the viability of the proposal to mitigate the communication straggler's drawbacks.

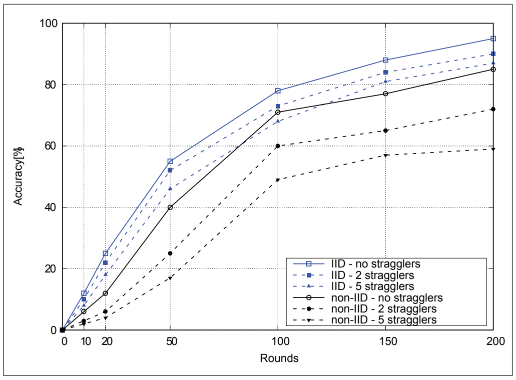

FIGURE 4. Accuracy vs. number of rounds for different number of stragglers.

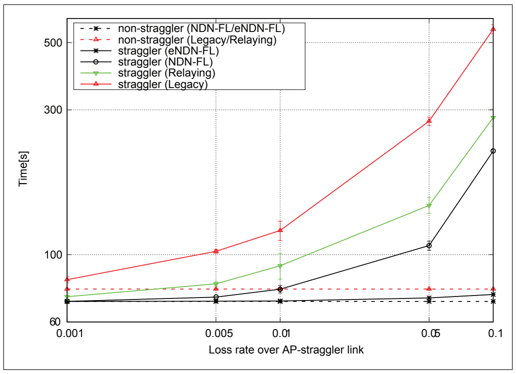

FIGURE 5. Training round time vs. packet loss rate over the AP-straggler link.

For all the considered networking solutions, we distinguish the average training round time of stragglers and non-stragglers, that is, clients affected by varying packet loss rate and clients that do not experience communication issues, respectively.

For T_{max} values lower than 600s, some stragglers can be excluded by the training round with a consequent reduction in the accuracy. In these cases, the advantages of our proposal are evident in Table 1, reporting the achieved accuracy by each compared solution for different T_{max} and loss conditions. TCP-based solutions are adversely impacted by worsening the channel conditions and reducing T_{max}.

Conclusion

In this article, we have proposed *eNDN-FL*, an NDN-based solution for FL that reduces the training round duration when stragglers are involved in the training procedures. *eNDN-FL* paves the way for FL client selection schemes that do not exclude stragglers and, instead, may benefit from their presence in terms of higher accuracy, without significantly affecting the training convergence. Synergies of the proposal with compression techniques as well as with semantic communications to further

Loss rate	Legacy		Relaying		NDN-FL		eNDN-FL	
	T_{max} = 300s	T_{max} = 120s	T_{max} = 300s	T_{max} = 120s	T_{max} = 300s	T_{max} = 120s	T_{max} = 300s	T_{max} = 120s
0.001	85%	85%	85%	85%	85%	85%	85%	85%
0.005	85%	85%	85%	85%	85%	85%	85%	85%
0.01	85%	**59%**	85%	85%	85%	85%	85%	85%
0.05	85%	**59%**	85%	**59%**	85%	85%	85%	85%
0.1	**59%**	**59%**	85%	**59%**	85%	**59%**	85%	85%

TABLE 1. Accuracy achieved when varying the training round deadline (T_{max}) for the compared schemes in the presence of 5 stragglers (out of 10 clients), experiencing varying loss rates.

reduce the network pressure will be also a subject matter of future works.

Acknowledgment

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" (PE00000001 — program "RESTART").

References

[1] Z. Zhao *et al.*, "Towards Efficient Communications in Federated Learning: A Contemporary Survey," J. Franklin Institute, 2023.
[2] T. Nishio and R. Yonetani, "Client Selection for Federated Learning With Heterogeneous Resources in Mobile Edge," *Proc. IEEE ICC*, 2019, pp. 1–7.
[3] Z. Lin *et al.*, "Relay-Assisted Cooperative Federated Learning," *IEEE Trans. Wireless Commun.*, vol. 21, no. 9, 2022, pp. 7148–64.
[4] Z. Wang *et al.*, "Federated Learning via Intelligent Reflecting Surface," *IEEE Trans. Wireless Commun.*, vol. 21, no. 2, 2021, pp. 808–22.
[5] M. Fu *et al.*, "Federated Learning via Unmanned Aerial Vehicle," *IEEE Trans. Wireless Commun.* (Early Access), 2023.
[6] D. Saxena *et al.*, "Named Data Networking: A Survey," *Computer Science Review*, vol. 19, 2016, pp. 15–55.
[7] X. Li *et al.*, "Advancing Software-Defined Service-Centric Networking Toward In-Network Intelligence," *IEEE Network*, vol. 35, no. 5, 2021, pp. 210–18.
[8] M. Amadeo *et al.*, "Client Discovery and Data Exchange in Edge-Based Federated Learning via Named Data Networking," *Proc. IEEE ICC*, 2022, pp. 2990–95.
[9] J. Lee et al., "Data Distribution-Aware Online Client Selection Algorithm for Federated Learning in Heterogeneous Networks," *IEEE Trans. Vehicular Technology*, vol. 72, no. 1, 2022, pp. 1127–36.
[10] G. Cleland *et al.*, "Fedcomm: Understanding Communication Protocols for Edge-Based Federated Learning," *IEEE/ACM UCC*, 2022, pp. 71–81.
[11] Y. Xiao *et al.*, "Time-Sensitive Learning for Heterogeneous Federated Edge Intelligence," *IEEE Trans. Mobile Computing*, vol. 23, no. 2, Feb. 2024, pp. 1382–1400.
[12] A. A. Abdellatif *et al.*, "Communication-Efficient Hierarchical Federated Learning for IoT Heterogeneous Systems With Imbalanced Data," *Future Generation Computer Systems*, vol. 128, 2022, pp. 406–19.
[13] S. R. Pandey *et al.*, "Goal-Oriented Communications in Federated Learning via Feedback on Risk-Averse Participation," arXiv preprint arXiv:2305.11633, 2023.
[14] A. Krizhevsky *et al.*, "Learning Multiple Layers of Features From Tiny Images," 2009.
[15] K. He *et al.*, "Deep Residual Learning for Image Recognition," *Proc. IEEE Conf. Computer Vision and Pattern Recognition*, 2016.

Biographies

Marica Amadeo is an assistant professor of telecommunications at the University of Reggio Calabria, Italy. Her main research interests are in the field of edge computing and future Internet architectures.

Claudia Campolo (claudia.campolo@unirc.it) is an associate professor of telecommunications at the University of Reggio Calabria, Italy. Her main research interests are in the field of vehicular networking and future Internet architectures.

Antonella Molinaro is a full professor of telecommunications with the University Mediterranea of Reggio Calabria, Italy, and Université Paris-Saclay, France. Her current research interests include 5G/6G networks, connected vehicles.

Giuseppe Ruggeri is an associate professor of telecommunications at the University of Reggio Calabria, Italy. His current interests include self organizing networks, IoT, Social IoT.

Gurtaj Singh is a Ph.D. student at the University of Reggio Calabria, Italy. His research fields are related to distributed ML, edge computing.

Open access funding provided by University of Reggio Calabria within the CRUI CARE agreement.