Computer Networks 267 (2025) 111353

Contents lists available at ScienceDirect

Computer Networks

journal homepage: www.elsevier.com/locate/comnet

Improving communication performance of Federated Learning: A networking perspective

Marica Amadeo [a,c], Claudia Campolo [b,c,*], Giuseppe Ruggeri [b,c], Antonella Molinaro [b,c,d]

[a] Università degli Studi di Messina, Italy
[b] Università degli Studi Mediterranea di Reggio Calabria, Italy
[c] CNIT, Consorzio Nazionale Interuniversitario per le Telecomunicazioni, Italy
[d] Université Paris-Saclay, CNRS, CentraleSupélec, France

ARTICLE INFO

Keywords:
Federated Learning
Host-centric communications
Information Centric Networking
Transport protocols

ABSTRACT

Federated Learning (FL) is gaining momentum as a promising solution to enable the efficient and privacy-preserving distributed training of Machine Learning (ML) models. Unlike centralized ML solutions, only the ML model and its updates are transferred between the clients and the aggregator server, eliminating the need to share large datasets. Notwithstanding, poor connectivity conditions experienced over the path that interconnects the FL clients and the aggregator server, either due to (wireless) channel losses or congestion, may deteriorate the training convergence. Several methods have been devised to reduce the training duration, primarily by minimizing data transfer through the design of ML algorithms at the application level. However, these solutions still exhibit unsettled issues, as they may only reduce the communication footprint but do not improve the communication process as a whole. Differently, in this work, our aim is to improve FL data exchange from a networking perspective by promoting Information Centric Networking (ICN) approaches rather than host-centric TCP/IP-based solutions. To this aim, we analyze the impact that host-centric transport protocols as well as ICN approaches have on the FL performance, in terms of duration of the model training and exchanged data (model and updates) load, under different channel loss settings. We show that ICN-based FL solutions significantly reduce the network data load and decrease the duration of the training round by up to an order of magnitude for high channel loss rates.

1. Introduction

The huge amount of data collected and generated by smartphones, laptops, and Internet of Things (IoT) devices has led to a surge of interest in Machine Learning (ML) across many application domains, ranging from augmented reality to industrial automation and autonomous driving. However, traditional centralized ML schemes, which require data to be transferred and processed in a central entity, can raise privacy and communication concerns. As a solution, Federated Learning (FL) has been proposed by Google to enable ML models to be executed in a distributed manner [1,2]. Instead of sharing their own private dataset, in FL multiple distributed clients act under the coordination of an aggregator server and locally train a shared ML model. The local training results, in the form of gradients or model parameters, are sent to a server, which aggregates them and shares the resulting global model with the clients for the next training round. The process continues iteratively for multiple rounds, potentially hundreds, typically until the requested accuracy is reached.

Model aggregation can be performed either at the edge or in cloud. Edge-based FL performs global model aggregation at the edge server, whereas cloud-based FL performs global model aggregation at the remote cloud. Edge-based FL is targeted in this work since it particularly suits applications demanding context-awareness and low-latency interactions with the clients [3].

FL can be deployed in two primary settings: siloed FL and cross-device FL, each tailored to different collaboration scenarios and types of participants [4]. Siloed FL operates in environments with a small number of participants, typically organizations or institutions, referred to as data silos. These silos are often data-rich entities that are almost always available, such as hospitals, banks, or research institutions that collaboratively train models without sharing sensitive data. Cross-device FL, on the contrary, involves collaboratively training a model across a large number of personal devices, such as smartphones, IoT or edge devices. They typically have intermittent connectivity, limited

* Corresponding author at: Università degli Studi Mediterranea di Reggio Calabria, Italy.
E-mail address: claudia.campolo@unirc.it (C. Campolo).

https://doi.org/10.1016/j.comnet.2025.111353
Received 9 January 2025; Received in revised form 7 April 2025; Accepted 2 May 2025
Available online 18 May 2025

computational power, and variable availability. In this paper, we consider the cross-device FL setting, where communication is often the primary bottleneck.

In fact, even if with FL the overall communication cost of exchanging model updates is lower than sharing large amounts of datasets from the devices in a centralized training setting, it is critical to save the communication bandwidth even further to make FL more efficient. Participating clients, especially in cross-device designs, can be mobile and have unreliable network connectivity on the error-prone bandwidth-limited wireless medium that connects them to the server, resulting in delays in the exchange of models and updates, and leading to disruption in FL performance [2].

So far, several works have specifically targeted communication issues in FL and strategies to deal with them [5–7]. For example, the works in [8–10] survey mechanisms such as data compression to reduce communication costs and overhead in FL. Various client selection mechanisms [11,12] have been proposed that exclude the so-called *communication stragglers*, i.e., those clients experiencing poor connectivity, failing to promptly share their model updates, with consequent delays in the aggregation procedure. However, this may result in a decrease in the amount of training data collected and a consequent slowdown in FL convergence [2]. Furthermore, most literature works focus on single-hop client–server wireless communication [13,14], although in realistic environments the aggregator can be hosted several hops away from the client. In such cases, in addition to the well-known limitations of the wireless medium, congestion can also occur due to the presence of other traffic flows over the multiple links that connect clients and the server, further challenging FL data exchange [15–17].

In summary, the FL convergence rate can be highly affected by the communication quality experienced throughout the entire (potentially multi-hop) path connecting clients and the aggregator server [18]. Hence, it is crucial to extensively dissect the role of network and transport protocols and identify solutions to improve FL communication performance, which is the focus of this paper.

Communication between FL clients and the server has typically been host-centric and based on the Internet Protocol (IP). At the transport layer, both the Transmission Control Protocol (TCP) and the User Datagram Protocol (UDP) can be used over IP. On the one hand, TCP guarantees a reliable exchange of the global model and of its updates after training from clients [19,20], but it may introduce non-negligible delays and overhead due to enforced retransmissions in error-prone channels with mobile clients. On the other hand, to support low-latency communications, some work considers the low-overhead and unreliable UDP [21], which however can be detrimental in terms of accuracy, since model updates from poorly connected clients can be lost.

More recently, a departure from the host-centric communication model has been devised and information-centric approaches have been introduced in FL. By leveraging name-based primitives, native multicasting and in-network caching, Information Centric Networking (ICN) provides efficient and effective data distribution, even in challenging environments [22]. The benefits of ICN in terms of communication performance for FL have been pioneeringly investigated in [23], while in [24] ICN is considered a solution to improve anonymity and privacy for participants in the FL process throughout all the steps of the training procedure. The naming scheme, the packet forwarding and the caching protocol are defined accordingly. There, all the communication components that produce overhead, including header fields and encompassing notification and encryption messages, are considered in the assessment of the overall communication efficiency of NDN compared to the considered benchmarks. In [25] the focus is on in-network client discovery and selection through ICN. There, the strengths of ICN compared to application-level solutions are theoretically discussed.

Nevertheless, to the best of our knowledge, in the literature there is no comprehensive and quantitative analysis of FL performance from a networking perspective, with focus on the transport layer dynamics, which considers traditional host-centric TCP/IP vs cutting-edge ICN approaches.

In order to fill this gap, in this work we aim to provide the following main original contributions:

- To dissect the main communication-related issues affecting the FL performance and relevant solutions devised in the literature, at different layers, along with relevant potential benefits and drawbacks.
- To analyze the impact on the FL performance of different host-centric transport solutions, selected among those that are commonly practically deployed in legacy TCP/IP-based implementations. Unlike previous works in the literature, *transport layer features* are specifically considered, both for the baseline TCP/IP solutions as well as for the NDN-based approach, which can directly affect scalability, reliability, and efficiency requirements of next-generation FL systems.
- To understand if and to what extent revolutionary information-centric approaches can improve FL performance, compared to host-centric approaches.

A performance evaluation has been conducted through large-scale extensive simulations in a *realistic network simulation framework*, ndnSIM [26], able to *accurately capture all transport layer dynamics, in presence of packet losses and/or under congestion conditions*. Achieved results show the superiority of ICN against host-centric solutions in terms of reduced training round time and exchanged traffic.

The remainder of the work is organized as follows. Section 2 describes the FL paradigm, discusses the relevant communication issues, and the main solutions in the literature. The host-centric and information-centric approaches for FL are theoretically discussed in Sections 3 and 4, respectively. Section 5 presents the conceived NDN framework to enable data exchange in FL. A comprehensive performance evaluation with the ndnSIM simulator is reported in Section 6. Section 7 concludes the work by providing hints on future work.

2. Federated learning: background, challenges and solutions

2.1. Basics

An FL system has two main kinds of entity, i.e., a server, acting as an FL aggregator, and a set of distributed clients. In cross-device design, these latter include a large number of mobile devices or edge nodes, with typically heterogeneous and limited network availability and computing capabilities. The relevant FL workflow is detailed in the following and is graphically sketched in Fig. 1. The server running the FL application selects a set of clients, among those that can potentially contribute to the training, randomly or according to a given policy [11], e.g., accounting for available computation capabilities and/or network conditions or data quality [12,27–29]. The server shares the global model and training settings with the selected clients, which locally train the model on their own dataset and, after completion, send the model updates to the server. If synchronous FL is considered, only after receiving updates from all selected clients, the server performs the model aggregation and disseminates the updated global model to the clients. The mentioned steps are re-iterated until a given number of rounds is reached and/or the targeted accuracy is achieved.

2.2. Communication issues and solutions

Since FL may consume significant computation and network resources at each client, a key target is to achieve high *time-to-accuracy* performance and therefore to reach model convergence in a reasonable (possibly short) time. The convergence of an FL task highly depends *(i)* on the time taken by the clients to perform the local training, according to their computing capabilities, and *(ii)* on the time needed to exchange

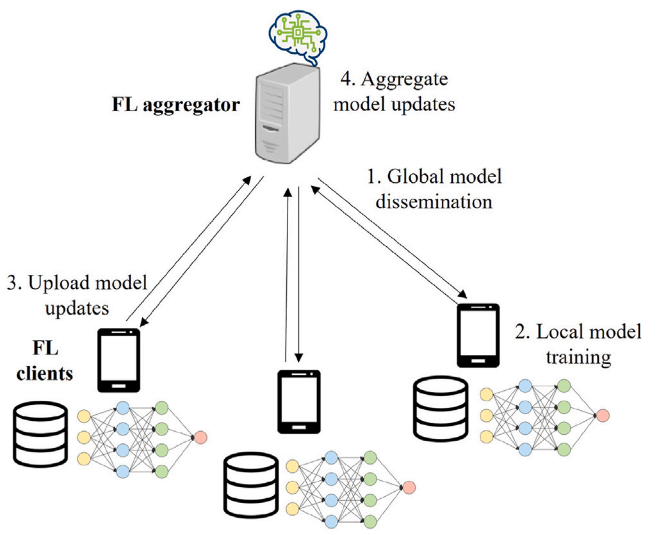


Fig. 1. FL players and steps: FL clients receive the global model from the FL aggregator, locally train it and upload the model updates to be aggregated.

FL data, that is, the global model and its updates, which depends on the communication capabilities of the links interconnecting clients and aggregator. In this section, we focus on the issues and relevant solutions to reduce the time contributions related to the exchange of the global model and of its updates. In fact, communication limits are primary factors that must be considered when designing FL systems [10].

Communication performance can significantly vary from client to client due to different wireless access technologies (e.g., fourth generation, 4G, vs. fifth generation, 5G), client locations (indoor vs. outdoor), and client–server channel conditions (clean vs. interfered). In the presence of hundreds of heterogeneous clients, the network bandwidth may even exhibit an order-of-magnitude difference [11]. In addition, communication stragglers, suffering from intermittent or unreliable connectivity that causes multiple packet losses recovered by forcing retransmissions, can slow down the overall training procedures. This is especially detrimental in the case of synchronous FL, when the server must wait for the slowest client to upload its model update. In the presence of a large number of clients, almost simultaneously uploading their model updates, the server can become a bottleneck, especially if multiple retransmissions are enforced.

Various mechanisms, acting at different layers and through different methodologies, have been proposed in the literature to reduce the training round duration and speed up the data exchange between clients and the server. Some of them are briefly overviewed in the following and summarized in Table 1.

ML-based solutions. The ML community has proposed techniques that try to act on the data, models and information managed by learning algorithms. They encompass, among others, quantization, sparsification, knowledge distillation [2,30,31]. More in detail, quantization strategies provide a low precision representation of weights, gradients or activations to reduce the total number of bits transmitted in each model update and thus, reduce the incurred communication footprint and latency transfer [32]. Sparsification techniques prevent irrelevant updates from being transmitted by, for example, removing redundant information and only transmitting the important values from local estimates. Knowledge distillation techniques can be applied in FL to send soft-label predictions, instead of heavier updated models or gradients [33]. However, such solutions require substantial changes to applications, since they are directly implemented at the algorithmic level of FL [43], and may unpredictably hurt the accuracy performance of models [31,44].

Time-budgeted FL. To limit the server waiting time in each training round, a deadline T, also known as *report window* [34], is typically considered [12,35], and set equal to the maximum tolerable training round time. If the model update from a client c_i is not received by the aggregator before the deadline, then the contribution from c_i is excluded from the FL aggregation. In practice, many approaches set an empirical report window, e.g., 2 min when considering a virtual keyboard search suggestion [34], and very few works investigated the impact of varying this parameter. The study in [12] shows how different deadlines, from 1 min to 10 min, affect the performances of trained models. The authors find that, although longer deadlines involve numerous clients in each round, the resulting performances are limited due to the smaller number of aggregation steps within the final deadline. On the other hand, shorter deadlines limit the number of clients contributing in each round with a reduction in the final accuracy, thus penalizing all the clients. Also, excluding late clients, while they are still training or transmitting the updates, lead to a waste of their computing resources and inefficient network usage.

In [28], the authors consider a fixed developer-preferred report window T, but also assume that the value can be extended in case the statistical utility of a client outweighs its slow speed. The Fedbalancer proposal in [36] computes, per each client c_i, the estimated completion time, t_{c_i}, according to the average uplink and downlink speed and batch training latency. Then, it selects the deadline with the best *deadline efficiency* (DDLe) computed as the ratio between the number of clients expected to complete before T, and T. A similar approach is adopted in [37], where the deadline is dynamically set in each round by considering the performance discrepancy among heterogeneous clients. The evaluation demonstrates that properly tuning the deadline is a critical factor that can significantly affect accuracy performance.

Relay-based solutions. To enhance communication opportunities, some solutions from the networking community leverage the presence of neighboring nodes [38,39], which act as relays between the clients and the aggregator. While this approach can improve connectivity, selecting such nodes may introduce additional complexity. In [40], an Unmanned Aerial Vehicle (UAV) serves as the FL aggregator and dynamically adjusts its communication distance from selected clients to enhance channel quality and mitigate communication stragglers. Nonetheless, this method may be impractical in realistic FL scenarios involving a potentially vast number of distributed clients.

AirComp-based solutions. By leveraging the waveform superposition property of multiple-access wireless channels, over-the-air computation (AirComp) enables fast model aggregation in FL [41]. However, over-the-air aggregation requires precise channel estimation and synchronization of clients, which becomes particularly challenging under mobility and highly dynamic channel conditions. To address these issues, Reconfigurable Intelligent Surfaces (RISs) are utilized in [42] to enhance model aggregation when AirComp is employed. RISs mitigate the detrimental fading effects that impair the merging of concurrent data transmissions from multiple clients. Despite their high potential for fast and reliable model aggregation, the design and deployment of RISs still present significant challenges: optimizing a large number of reconfigurable phase shifts in rapidly varying channel environments adds substantial complexity.

Conclusive remarks. The aforementioned approaches share the common objective of targeting an efficient FL implementation, by reducing the amount of exchanged data traffic in the network and/or accelerating the data exchange. They achieve such a goal through additional data processing at the application or physical layers or by involving additional network nodes (to be discovered and properly selected) in the FL process, all without modifying the existing networking protocols. On the other hand, in this work, we demonstrate how adopting a novel networking approach, specifically designed for effective and efficient data dissemination in both wired and mobile wireless scenarios, can directly impact both the volume of exchanged data and the speed of delivery, thereby further enhancing the FL process.

Table 1
Literature overview.

Category	Layer	Main idea	Potential drawbacks	Exemplary works
ML-based solutions	Application	Reduce the amount of data, models and information	Substantial changes to applications; accuracy degradation	[2,30–33]
Time-budgeted solutions	Application	Set a deadline within which model updates should be received before the aggregation step	Difficult tuning of the deadline	[12,28,34–37]
Relay-based solutions	Application	Additional nodes closer to the aggregator than the selected clients forward model (update)	Discovery and selection of relays; Deployment costs (e.g., for UAVs);	[38–40]
AirComp-based solutions	Access	Merge the concurrent data transmission from multiple clients and performs "over-the-air" data aggregation	Channel estimation and synchronization of clients, challenging under mobility and highly dynamic channel conditions	[41,42]

3. Host-centric communications for FL

3.1. Basics: TCP- vs. UDP-based solutions

In most FL implementations, data exchange between clients and the server relies on the TCP/IP protocol suite, typical of Internet communications. This applies whether simple applications use TCP directly (e.g., [19,20]), or optimized application-layer protocols are built on top of TCP (e.g., [45,46]). The reason for this widespread adoption is that TCP, with its connection-oriented, reliable, ordered, and error-checked delivery, offers the necessary guarantees to safely transport FL data, making it a generally suitable choice.

To speed up the delivery process, some work considers the connectionless UDP protocol, which does not provide any mechanisms for flow control, error recovery, or congestion control. In particular, the work in [43] presents a comparison between the reliable TCP and the unreliable UDP when running an FL application. The authors consider a 4G and a Wi-Fi network scenario, and find that TCP outperforms UDP with higher accuracy and shorter communication times. In contrast, when considering low-bandwidth 3G connectivity, UDP outperforms TCP in terms of lower communication time, but accuracy is inconsistent due to high packet losses. The authors observe that, thanks to its reliability, TCP preserves accuracy in all considered network conditions by retransmitting unsuccessful packets that carried model updates from clients and were incorrectly received or not successfully transmitted. However, retransmissions imply further delays that could not be tolerated by latency-critical FL applications.

The work in [47] presents an optimized UDP-based FL scheme, where clients implement forward error correction (FEC) and retransmissions to deliver model updates in lossy channels. However, the conceived techniques increase the signaling overhead, thus limiting the speed of the UDP protocol. The work in [48] departs greatly from TCP by proposing the Loss tolerant Transmission Protocol (LTP). LTP allows for partial data loss (loss-tolerant transmission), through the use of out-of-order transmission and out-of-order acknowledgments (ACKs) to counteract non-congestion packet loss and the many-to-one "incast" traffic patterns, which characterize distributed ML training with a centralized server. Although conceived to be in principle transparent to the ML developers, it entails to be developed from scratch on top of UDP and some modifications in the client–server sockets are required.

In summary, TCP is the default choice for FL applications where reliability and accuracy are of concern. Therefore, in the remainder of this section, we will focus on TCP-based solutions only.

3.2. Potential TCP performance improvements

Regardless to its application in FL settings, to improve the TCP performance in challenging environments, the following main solutions have been investigated in the literature: *(i)* congestion control algorithms specifically designed for error-prone channels or connections; *(ii)* mechanisms for explicit congestion notification; *(iii)* multi-path transport solutions when multiple network interfaces are available at the end-hosts.

Advanced congestion control. Various versions of TCP have been deployed over the years, with the main objective of improving the congestion control algorithm. For example, among the latest proposals implemented by default in many systems, TCP Binary Increase Congestion control (BIC) [49] and CUBIC [50] adopt a reactive scheme that detects congestion after it has actually occurred, using packet losses as evidence of congestion. BIC, the default TCP variant on Linux systems until 2006, uses a binary search approach to adjust the congestion window size and tries to maintain a balance between aggressive and conservative adjustments. CUBIC, the new default TCP variant on Linux systems since November 2006 (hence, the default in most Android devices), uses a cubic function to control the congestion window size. However, it is well known that TCP performs poorly in the case of unreliable connectivity: losses due to errors over the wireless channel are mistakenly considered as a congestion signal, thus forcing the source to reduce the transmission rate, with a negative impact on the achieved throughput. One of the most popular congestion control solutions, specifically designed to handle error-prone wireless links, is TCP Westwood [51]. The algorithm adjusts the sender transmission rate and, therefore, the congestion control parameters, according to the actual estimated end-to-end bandwidth. The so-called eligibility rate is obtained by constantly monitoring the rate of new received ACKs and duplicated ACKs (DUPACKs). A packet loss is detected at the reception of 3 DUPACKs or at the timeout expiration. The extension TCP Westwood Plus [52] incorporates several optimizations to handle overestimation of bandwidth in the presence of congestion and improve performance [53]. It includes an adaptive Retransmission Timeout (RTO) computation to better handle data retransmissions based on the observed network conditions.

Explicit congestion notification. To proactively manage congestion and help distinguish packet loss caused by network congestion from losses due to channel errors or interference, Active Queue Management (AQM) with Explicit Congestion Notification (ECN) has been introduced [54]. AQM actively monitors the queue and signals congestion by marking packets rather than dropping them. When combined with ECN, this approach enables the network to notify endpoints of impending congestion by marking packets at the IP layer. This allows terminals to adjust their sending rates and avoid packet loss due to congestion, improving overall network performance.

Multi-path transport. To benefit from multiple available connection opportunities, typically available in today's end-devices, multi-path TCP (MPTCP) has been proposed in [55]. MPTCP allows (multi-homed and multi-addressed) end-hosts to simultaneously transmit a single data stream across a number of TCP sessions, thus improving the throughput and supporting resiliency in case one subflow experiments packet losses. A single MPTCP connection is first established with the counterpart, similarly to a regular TCP connection. If extra paths are available, additional TCP sessions (referred to as subflows) are created on these paths and combined with the main session, which still appears as a single logical connection to both end-host applications. At any time during an MPTCP connection, the end-host can announce additional addresses on which it can be reached or remove existing ones.

Conclusive remarks. Despite the potential benefits, MPTCP exhibits several challenges that hinder its widespread adoption in practice [55,56]. Indeed, MPTCP introduces additional complexity and signaling overhead to maintain TCP subflows and coordinate data transmission across different paths. Moreover, it may face compatibility issues with legacy network infrastructure and middleboxes, which could block subflows. Conversely, congestion control algorithms like Westwood are implemented at the sender-side and can typically co-exist with other TCP implementations, without requiring changes to network infrastructure or at the receiver-side. This is why, in the following, we will focus our analysis on the impact of TCP congestion control on FL applications. In doing so, we advance the state-of-the-art which typically fails to deepen the analysis of different TCP protocol versions in the FL context. We also incorporate AQM/ECN into our analysis, as it is particularly beneficial for improving the performance of TCP-based communications.

4. Information-centric communications for FL

4.1. Basics: Named data networking

ICN is a communication paradigm for the future Internet that directly focuses on content names instead of IP addresses. In the following, we refer to the Named Data Networking (NDN) architecture [57], one of the most prominent ICN implementations.

NDN communication is connectionless and based on two packet types: *Interest*, transmitted by clients/consumers to retrieve contents by name, and *Data*, transmitted by *any* node owning the requested contents. Data packets are self-consistent units that can be potentially cached by any node in the network. Therefore, contents can be retrieved from multiple providers, i.e., the original source or a caching node, and through multiple paths.

In particular, as illustrated in Fig. 2, an NDN node maintains three data structures in the forwarding plane: *(i)* a Content Store to cache incoming Data packets; *(ii)* a Pending Interest Table (PIT) to temporarily store incoming Interests and *(iii)* a Forwarding Information Base (FIB) to identify the forwarding interfaces toward content sources. At the Interest reception, an NDN node first checks the Content Store to find a matching Data. If this lookup fails, the node checks the PIT. If the PIT contains a request pending for the same Data, then the node updates the existing PIT entry by recording the additional incoming interface of the Interest, and discards the packet. By doing so, Interests carrying the request for the same content are aggregated, and redundant transmissions to the data source are avoided. If the PIT matching fails, then the node creates a new PIT entry and looks into the FIB to find outgoing interface(s) for the request.

Data packets simply follow the chain of PIT entries back to the consumers, and they can be replicated over multiple outgoing links according to the interfaces recorded in the PIT during the Interest forwarding. As a result, multicast delivery is inherently supported by NDN. Unsolicited Data packets, i.e., contents without a matching PIT entry, are discarded.

Depending on the implemented forwarding strategy, single-path or multi-path Interest forwarding can be enabled [59]. Indeed, the same Interest can be transmitted over the best single path, or in parallel over multiple interfaces. The same node can even apply different forwarding strategies based on the content type. Moreover, Interests that are not consumed by the Data packets within the expected round-trip-time (RTT) can be retransmitted by intermediate nodes.

4.2. Why NDN for FL

NDN can improve FL communication performance through the following features:

- *In-network caching.* In-network caching at intermediate nodes facilitates data exchange between the FL aggregator and clients by ensuring that the global model and/or model updates can be retrieved in a reliable and efficient manner even in the case of packet losses. This is particularly beneficial in cross-device FL environments where clients are connected through error-prone wireless links, as it minimizes the need for retransmissions from the source and improves the robustness of the overall communication process.
- *Request aggregation.* The ability of NDN to aggregate identical requests (Interests) for the same data from different consumers prevents the network from being overwhelmed with redundant transmissions. This is valuable in FL scenarios where multiple clients simultaneously request the global model. By avoiding duplicate requests, NDN reduces the load on potentially congested backhaul links and alleviates bottlenecks at the aggregator, ensuring a more scalable and efficient FL operation.
- *Native multicast data delivery.* NDN inherently supports multicast, potentially enabling a single transmission of the global model from the aggregator to serve all participating clients. This significantly reduces network resource usage, which is critical in FL setups where large global models must be distributed to many clients in the downlink direction.
- *Native support of client mobility.* Especially in cross-device FL, clients can be mobile (e.g., smartphones). Unlike traditional host-centric networks that rely on fixed IP addresses and require complex handover and tunneling mechanisms to maintain connectivity, NDN allows mobile clients to seamlessly reissue Interests from their new location. This eliminates the need for re-establishing connections or updating routing tables tied to specific addresses, simplifying the network management, and ensuring uninterrupted participation in FL tasks.

In our previous work in [23], we presented an NDN-based framework for FL that enables the discovery and selection of clients, as well as the data exchange between clients and the aggregator. There, the plain NDN implementation was considered for data (global model and model updates) exchange between the aggregator and the clients and compared against an application layer solution utilizing the Message Queuing Telemetry Transport (MQTT) protocol. We showed that the NDN implementation outperforms MQTT in terms of communication footprint, and the advantage is more remarkable for higher number of clients. Then, in [60], we defined a multi-path forwarding strategy to specifically counteract the communication straggler effect in FL settings, by leveraging NDN and customizing its forwarding fabric to improve model delivery in lossy channels. Our solution allows stragglers to participate in the training with advantages in terms of reduced training time and higher accuracy.

In this paper, we make a step forward with regard to our past contributions and focus our analysis on more advanced transport layer features, which can directly impact scalability, reliability, and efficiency

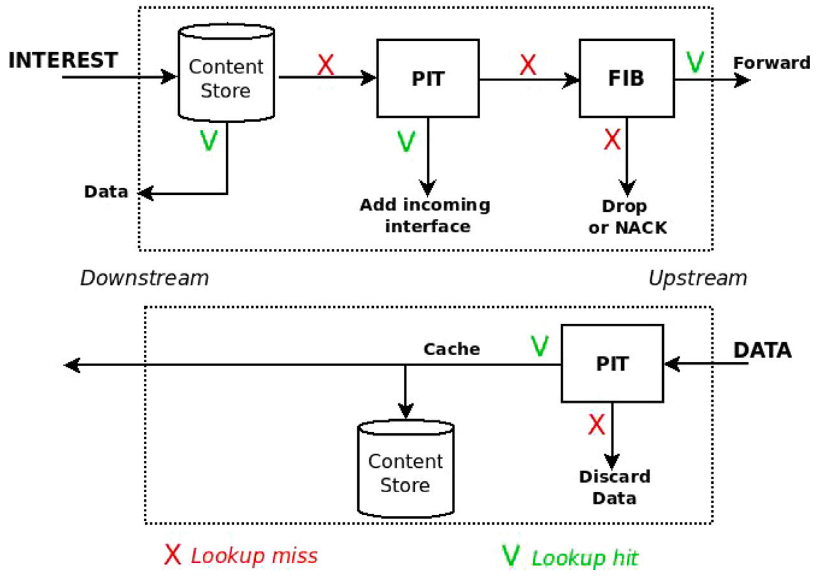


Fig. 2. Interest and Data packet processing in NDN.
Source: Adapted from [58].

requirements of FL systems. Transport layer dynamics are accurately analyzed in presence of packet losses and/or under congestion conditions, by accounting for end-to-end retransmissions, the only possible for existing TCP/IP-based solutions, as well as for retransmissions and caching by intermediate NDN nodes in the case of NDN.

4.3. Transport solutions for NDN implementations

Due to the peculiarities of NDN, such as in-network caching, multipath forwarding and multicast data delivery, traditional connection-oriented TCP congestion control algorithms cannot be straightforwardly applied in NDN networks.

A few solutions adapted to NDN appeared in the literature in recent years [61–64]. Among them, we consider PCON (Practical CONgestion control), proposed in [64], as a reference in this work because it is one of the most popular design available in the literature and also implemented in the NDN release [65]. With PCON, NDN nodes implement an hop-by-hop congestion control scheme: they detect congestion on their local links by using AQM, and signal this state to clients and downstream routers by explicitly marking Data packets. By doing so, on the one hand, clients react by reducing their sending rate; on the other hand, the downstream routers react by partially diverting subsequent Interests to alternative paths and/or passing the signal further downstream.

When loss is detected, NDN consumers can apply a classic loss-based TCP congestion control algorithm, like BIC and CUBIC, to reduce the Interest sending rate. In parallel, when the demanded transmission rate exceeds the capacity of the selected delivery path, i.e, typically the shortest one, NDN routers can implement multipath forwarding: Interests are incrementally distributed on the next shortest path(s). Since NDN implements connectionless delivery and Data packets are self-consistent units, multipath forwarding does not inherit the drawbacks of MPTCP.

4.4. NDN deployment options

By heavily departing from the conventional host-centric TCP/IP approach, NDN raises issues for its actual deployment in current data networks and entails some changes. Two distinct deployment approaches have been identified so far.

The *overlay* approach, over the existing Internet infrastructure, is typically preferred to be implemented on a large-scale, since it basically requires the (low-cost) implementation of interfaces between NDN and TCP/IP protocols [66,67] or the introduction of gateways for tunneling NDN communications over IP backbones [68].

Conversely, the *clean-slate* option, which is aimed to replace the TCP/IP protocol suite, is especially suited in greenfield environments, where hardware and software solutions are implemented from scratch in an easier manner than in other domains [69,70], like edge environments targeted in this work. The latter ones are supposed to cover small-to-medium sized areas (at most, a city area) with no more than a hundred of *network edge nodes* [71–74]. Therefore, in principle, it would be feasible to deploy NDN as a clean-slate solution and thus, benefiting from its features like in-network caching, for exchanging FL model parameters, and multicast and Interest aggregation to retrieve the global model.

In parallel, networking is generally evolving toward *softwarized designs* [75] and *virtualized network* functions [76], which facilitate the implementation of multiple protocol stacks over the same physical device, thus avoiding any additional hardware cost (i.e., devices may not necessarily need to be replaced). In this context, depending on the application, the NDN paradigm could co-exist with other networking solutions, e.g., the conventional IP-based ones.

Hence, deploying NDN is doable in current data networks and, especially, at the targeted network edge, provided that the aforementioned changes are enforced.

5. The FL-NDN reference framework

FL communications require push-based primitives, which necessitate customization of the legacy NDN operation, as NDN is inherently pull-based: a node can transmit a Data packet only after receiving a request (i.e., an Interest packet) for the corresponding content.

To enable push-style communication, we rely on *Interest Notifications*, i.e., Interest packets that do not retrieve Data packets but carry small, arbitrary information encoded in additional name components. Originally proposed in [77], Interest Notifications redefine the semantics of Interest packets to support lightweight signaling with minimal overhead. They have been successfully employed in several domains, including producer mobility management [78] and IoT applications [79,80].

In our design, we implement a signaling mechanism based on Interest Notifications that integrates seamlessly into the existing NDN

Table 2
Comparison of push-based NDN communication approaches.

Scheme	When	Message type	Sender(s)	Dissemination
Proposal	Client selection	Routing adv.	Clients	Controlled flooding
	Client selection	Routing adv.	Server	Controlled flooding
	Local training end	Int. Notification	Clients	Unicast
	Model aggregation end	Int. Notification	Server	Unicast
[24]	Local training end	Announcement	Clients	Broadcast
	Model aggregation end	Announcement	Server	Broadcast

routing framework, similarly to [60]. At the beginning of the FL process, during client selection, the aggregator and the clients exchange a common FL application prefix and agree upon a naming convention.

Specifically, global model names follow the structure `FL_app/server/round_i`, where `FL_app` identifies the application (e.g., `/FL/SpeechEmotionRecognition/anger/`), `server` refers to the aggregator node, and `round_i` indexes the training round. The server advertises this prefix via standard NDN routing (e.g., Named-data Link State Routing, NLSR [81]), restricted to the edge domain where the FL task is deployed, in order to populate the FIBs of intermediate routers and clients.

Names for model updates follow a similar structure: `FL_app/client_k/round_i`, where `client_k` uniquely identifies the FL client producing the model update. Clients, once joining the FL application, transmit the routing advertisement to populate intermediate FIBs with the name prefix of the updated model they will produce later.

At the first training round, clients download the global model with the regular NDN Interest/Data exchange. Then, they perform local training and, when the process completes, they send an Interest Notification using the prefix `FL_app/server/`, appending the suffix `client_k/round_i`. By leveraging the longest prefix matching in the FIBs, the Interest is forwarded toward the server, which constructs the full name of the model update (i.e., `FL_app/client_k/round_i`) and initiates its retrieval.

After receiving all model updates, the server performs aggregation and subsequently issues an Interest Notification to all clients using the prefix `FL_app/client_k/`, prompting them to download the updated global model.

The resulting workflow is illustrated in Fig. 3, where two clients, C_1 and C_2, after joining the FL application, start by downloading the global model in the first training round with regular Interest/Data packets.[1] Interests from C_1 are forwarded directly to the server, while those from C_2, which are slightly delayed, are aggregated in the PIT of the intermediate node, namely a Base Station (BS). Once the global model is retrieved, both clients perform local training and send their Interest Notifications upon completion. After collecting all model updates, the server initiates aggregation and sends an Interest Notification to enable the clients to retrieve the updated global model, thereby starting the next training round.

Using explicit announcements to trigger FL model exchange has been also proposed in [24]. However, unlike [24], which uses broadcast notifications, our design leverages NDN's native routing infrastructure to populate FIBs. By doing so, the routing protocol can identify optimal paths toward the aggregator for retrieving updates and toward clients for delivering the global model.

Differences between the two approaches which both overhaul NDN to support the push-based FL communication model are summarized in Table 2.

[1] For the sake of clarity, a single Interest/Data exchange is depicted in the Figure, but of course multiple Interest/Data packets are exchanged to collect the model.

Table 3
Comparison of host-centric TCP transport and NDN transport solutions.

Feature	Host-centric TCP transport	NDN transport
Communication Model	Connection-oriented	Connectionless
Data Retrieval	Based on host address	Based on named content
Forwarding	Stateless, host-to-host	Stateful, content-based
Scalability	Limited by the number of connections	Scales well with caching and Interest aggregation
Congestion Control	End-to-end, based on packet loss or delay	Hop-by-hop, Interest/data flow control
Reliability	Ensured through retransmissions by sender	Ensured by retransmissions by sender and intermediate nodes
Caching	Not inherently supported	Built-in in-network caching
Traffic Patterns	Unicast traffic	Unicast and multicast
Mobility Support	Limited, requires handover mechanisms	Intrinsic, as data is retrieved by name
Reference Approaches	TCP CUBIC [50], TCP BIC [49], TCP Westwood+[53]	PCON [64]

6. NDN vs TCP/IP based FL: Performance analysis

TCP/IP and NDN approaches fundamentally differ in their communication models, data retrieval mechanisms, and transport layer capabilities, which significantly impact their performance in FL scenarios. Table 3 compares the two solutions by highlighting their key features.

Theoretically, NDN appears to address several challenges inherent to FL, offering promising solutions for issues such as scalability, reliability and traffic optimization. In the following, we delve into a practical evaluation of NDN to quantify these theoretical advantages, assessing its actual benefits and comparing its effectiveness against that of TCP/IP in FL scenarios.

In particular, we evaluate the performance of the NDN-based approach against three popular TCP versions, namely CUBIC [50], BIC [49] and Westwood+ [52], which are representative of different congestion control mechanisms. CUBIC and BIC are representative of loss-based window control mechanisms, which use a packet loss as a congestion signal to reduce the sending rate. Westwood+ is instead specifically designed for error-prone wireless links and, in case of losses, it takes into account an estimate of the available bandwidth to properly set the sending rate.

We also consider the NDN approach implementing two variants of the PCON transport, i.e., BIC and CUBIC. In both TCP/IP and PCON/NDN approaches, we integrate the AQM/ECN mechanism.

6.1. Simulation settings and scenarios

Simulations have been conducted in ns-3 and, in particular, the NDN architecture has been realistically reproduced with ndnSIM [26], an ns-3 module deployed by the NDN community.

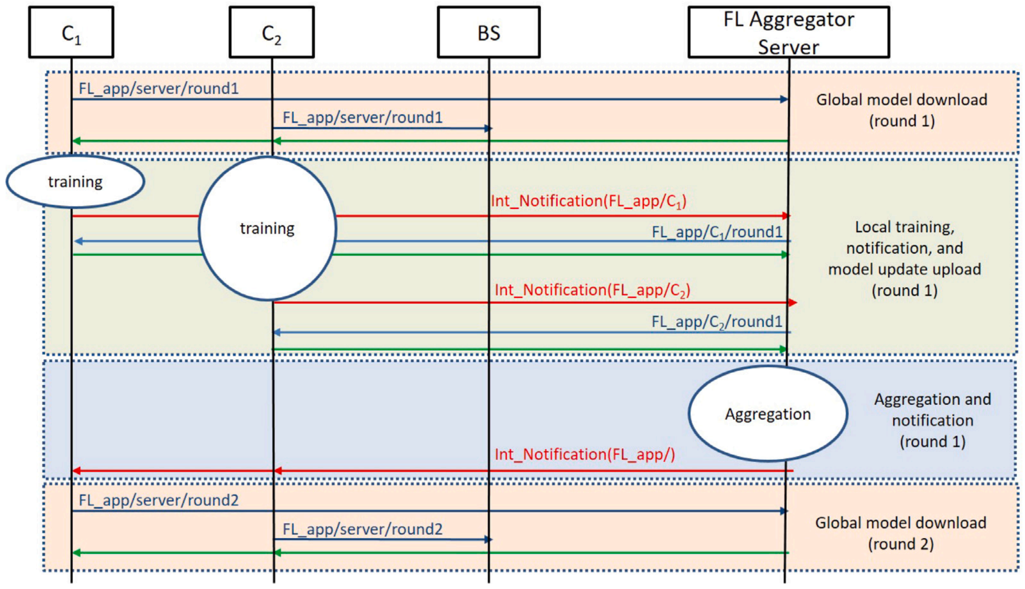

C₁
C₂
BS
FL Aggregator Server
FL_app/server/round1
FL_app/server/round1
Global model download (round 1)
training
training
Int_Notification(FL_app/C₁)
FL_app/C₁/round1
Local training, notification, and model update upload (round 1)
Int_Notification(FL_app/C₂)
FL_app/C₂/round1
Aggregation
Aggregation and notification (round 1)
Int_Notification(FL_app/)
FL_app/server/round2
FL_app/server/round2
Global model download (round 2)

Fig. 3. Reference workflow for FL-NDN: regular Interest packets are represented with blue arrows, regular Data packets are represented with green arrows, Interest Notifications are represented with red arrows.

We consider an object recognition learning task when using the CIFAR-10 dataset [82], which is widely adopted in the ML research community for simulations and testing. The used neural network (NN) model is MobileNet [83] and its size is 16MB. The global model and its updates, assumed of the same size of the model [84], are divided into 16,000 data packets, each carrying a payload of 1,000 bytes. Therefore, for both TCP/IP and NDN approaches, each client completes the download of the global model upon receiving all 16,000 data packets and starts the local training.

The local training time is drawn from a Gaussian distribution [12]. We consider two representative device classes: *(i) High-end smartphones*, modeled with a training time $\sim \mathcal{N}(\mu = 10\text{ s}, \sigma = 2\text{ s})$; and *(ii) Mid-range smartphones/tablets*, with training time $\sim \mathcal{N}(\mu = 20\text{ s}, \sigma = 4\text{ s})$ [83]. Each client is randomly assigned to one of these classes, thus introducing variability in the local training phase across the federation. In the uplink phase, the aggregator must collect all 16,000 data packets from each client before starting the per-round aggregation with the received model updates.

Without loss of generality, aggregation is performed according to the FedAvg [1], because it is one of the most commonly used techniques. Moreover, since our analysis is focused on communication aspects, whatever the considered networking solution, the aggregation algorithm would affect results in the same manner. Under the considered assumption, at a given round, r, the parameters of each client, i, are weighted and averaged to produce a global model $\omega^{r+1}_{i,glob}$:

$$\omega^{r+1}_{i,glob} \leftarrow \sum_{i\in\mathcal{N}} \frac{D_i}{D}\omega^{r+1}_i, \tag{1}$$

where the weight factor, $\frac{D_i}{D}$ (with $D = \sum_{i\in\mathcal{N}} D_i$), is the proportion of the client's data volume.

Two scenarios are simulated:

- *Chain topology*, depicted in Fig. 4(a), where a set of clients connected to a 5G BS participate in a synchronous FL training process, coordinated by an aggregator hosted on an edge server. An intermediate router connects the BS to the server via a wired, lossless link. To generalize the scenario, we also extend the topology by varying the number of hops between the FL aggregator and the clients from 3 to 10. The number of clients ranges from 50 to 300.
- *Hierarchical tree topology*: representing a larger scenario that mimics a hierarchical edge network topology, depicted in Fig. 4(b), where leaf nodes act as 5G BSs serving a variable set of mobile devices. The FL aggregator is co-located with the edge root node and connected to the BSs via multi-hop wired paths.

In both scenarios, we assume that the wireless links between FL clients and BSs are subject to a variable error rate that can be configured as an input simulation parameter.

6.2. Performance metrics

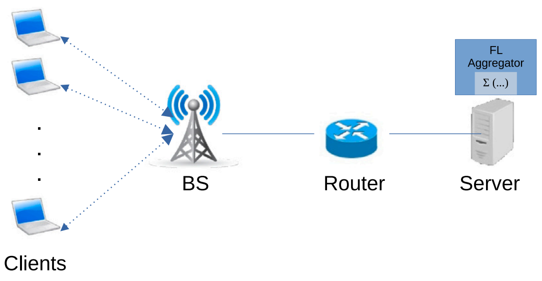
The following metrics are computed for performance comparison:

- *Training round time*, which is computed as the sum of four distinct contributions: *(i)* the time for a client to download the global model; *(ii)* the local training time; *(iii)* the time for a client to upload the model update; *(iv)* the aggregation time at the server.
- *Amount of data packets exchanged in a training round*. The metric accounts for all the packets (composing the FL model and its updates) transmitted to deliver the global model from the server to the clients, and all the packets transmitted to deliver the updated models from the clients to the server.
- *Number of hops in the model download phase*. This metric is computed in the tree topology scenario and captures the average number of hops traversed by data packets during the global model distribution. While TCP/IP-based approaches consider the model to be always transmitted by the aggregator, with the NDN solutions the model can be transmitted by intermediate caching nodes on the path between the aggregator and the clients.

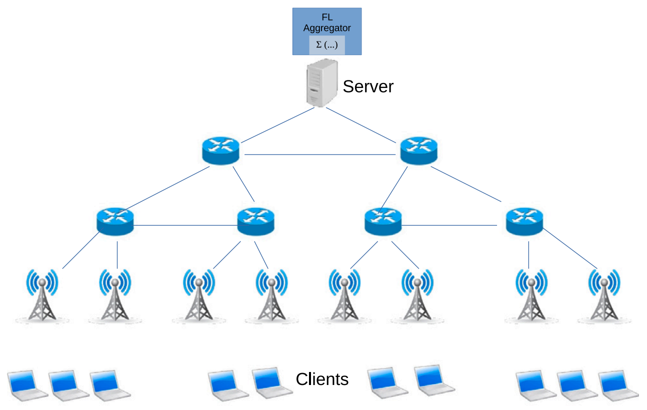
Results are averaged over 20 runs and reported with 95% confidence intervals.

6.3. Results in a chain topology

A preliminary evaluation is performed when running an FL application over wireless lossy channels in the chain topology in Fig. 4(a). The 5G downlink and uplink bandwidth limits between FL clients and the BS are set equal to 100Mbps and 50Mbps, respectively [85]. The error rate in the wireless links between clients and BS varies in the

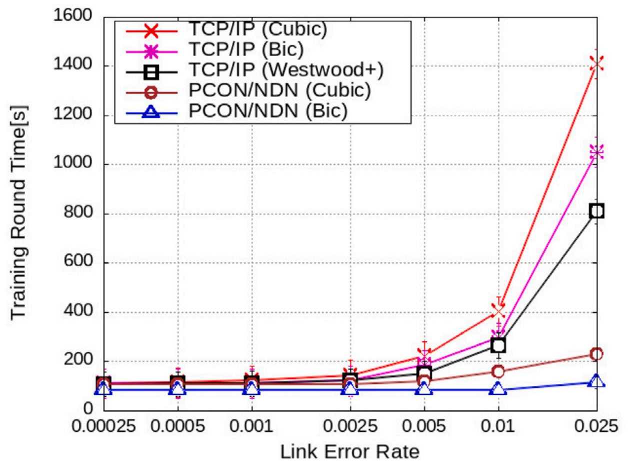


(a) Chain topology.

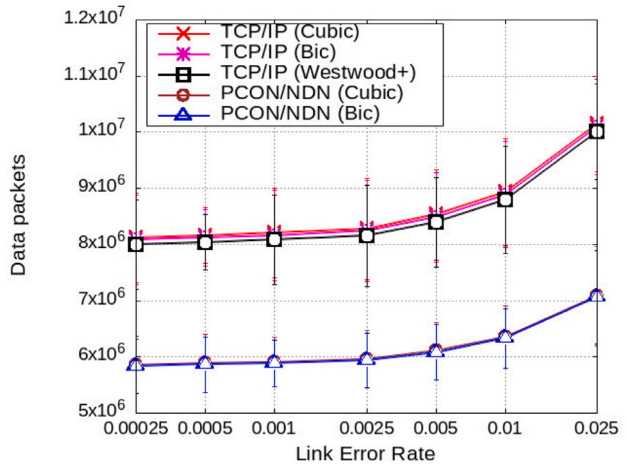


(b) Hierarchical edge network topology.

Fig. 4. Network topologies considered in simulations: FL clients and FL aggregator connected through a chain topology (a) and through a hierarchical edge network topology (b).



(a) Training round time.



(b) Number of data packets transmitted during one training round.

Fig. 5. Metrics of TCP/IP and NDN approaches in the presence of different congestion control algorithms (chain topology with three hops among FL clients and server).

range [0.00025 and 0.025]. The selection of the maximum error rate is due to the fact that, after this value, clients are not able to complete the training round within the (maximum) considered deadline, which is set to 30 min [36,86,87].

Fig. 5(a) shows the training round time when varying the link error rate and considering 50 clients. It can be observed that, for low error rates, there are no remarkable differences between the compared TCP- and NDN-based FL approaches. Conversely, when the link error rate increases, the performance becomes highly sensitive to the choice of the networking and congestion control approach. As expected, Westwood+ guarantees the lowest training round time among the TCP/IP versions, thanks to the implemented congestion control algorithm optimized for wireless communications. Such a finding suggests that a blind usage of the TCP protocols may lead to poor FL performance.

When the link error rate increases, the benefits of NDN become clearly evident, due to its natively implemented in-network caching and multicasting mechanisms, which enhance the dissemination of the global model and updates. Notably, PCON/NDN with BIC congestion control outperforms CUBIC. Similar results are observed with BIC and CUBIC in the TCP/IP approach. This is primarily because the simulated network has a limited RTT, which causes CUBIC to be less aggressive in exploring available bandwidth compared to BIC. Furthermore, in environments with more variable bandwidth availability, CUBIC's approach to congestion window growth is less adaptive than BIC's binary search, resulting in suboptimal performance.

Fig. 5(b) shows the average number of data packets transmitted during a training round. It can be observed that the performance is essentially the same for TCP/IP CUBIC, BIC and Westwood+. This is because the different congestion control algorithms affect the speed of packet delivery, but not the overall amount of exchanged traffic. Similarly, the performance of PCON/NDN with CUBIC and BIC is essentially the same. It can be observed that traffic increases with the link error rate, since more packets are lost due to channel errors and retransmissions are triggered. However, the number of transmitted packets with PCON/NDN is significantly lower than with TCP/IP, thanks to in-network caching and multicasting features, which accelerate delivery and aid in recovering from losses.

To better understand the impact of the distance between clients and the aggregator, Fig. 6 shows the number of data packets transmitted during a training round as the hop count increases from 3 to 10. A variable number of clients is considered, ranging from 50 to 300, while the link error rate is set to 0.0025. TCP/IP Westwood+ is compared against PCON/NDN BIC; other control mechanisms are not shown since no significant differences were observed, as previously explained. It can be observed that the number of transmitted data packets increases with the hop count, since each additional hop incurs additional transmissions per packet. Moreover, the gap between TCP/IP and NDN becomes substantially larger as the number of clients increases. In particular, in the 300-clients case, the difference in total packet transmissions between TCP/IP and NDN grows significantly with hop count. This effect is less pronounced for smaller numbers of clients (e.g., 50). All in all, the traffic reduction advantages offered by NDN are more pronounced when the number of clients and the distance between the client and aggregator increase. This is because data packets can be retrieved from the caches of intermediate network nodes rather than

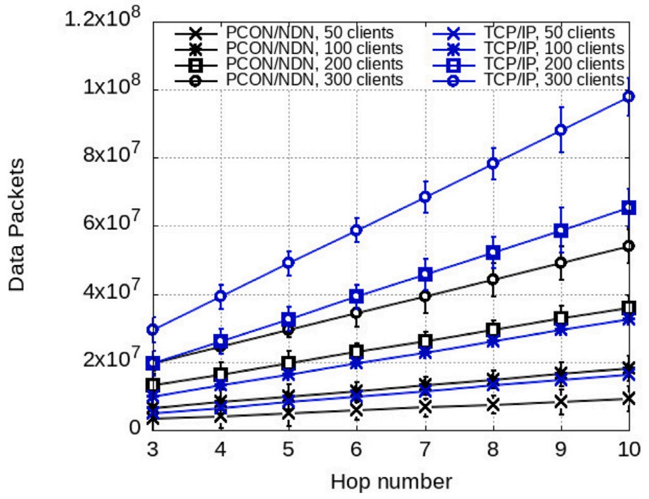


Fig. 6. Number of data packets transmitted during one training round, when varying the distance (i.e., hop number) between clients and the FL aggregator (chain topology).

being transmitted directly from the aggregator. Consequently, NDN is especially beneficial in cross-device FL scenarios, where a large number of (potentially heterogeneous) clients is expected.

6.4. Results in a tree topology

To compare the performance of host-centric and information-centric communication approaches in a larger scenario, we consider the hierarchical edge network topology in Fig. 4(b) and compare the best performing algorithms: Westwood+ and PCON/NDN with BIC. Among the available clients connected to the BSs, 100 clients are selected to participate in the FL application.

In Fig. 7, we report the FL performance in terms of duration of the training round, amount of exchanged data packets and number of hops traversed by data packets during the global model download phase. It can be observed the same trend, but with higher values, achieved in the chain network topology, with PCON/NDN outperforming TCP/IP, in terms of reduced training round time (Fig. 7(a)) and transmitted data traffic (Fig. 7(b)). Notably, the reduction in training round time becomes particularly significant as the link error rate increases.

To better understand the impact of NDN's in-network caching and multicast support, Fig. 8 shows the average number of hops traversed by data packets during the global model distribution. It can be observed that, in the TCP/IP case, each data packet must be transmitted by the source (i.e., the aggregator), resulting in a constant hop count of 4. In contrast, with PCON/NDN, the number of hops decreases as the link error rate increases. This occurs because lost packets are retrieved from intermediate nodes, rather than being retransmitted by the origin server. As a result, NDN proves highly advantageous in the presence of communication stragglers.

6.5. A closer look into traffic reduction with NDN

To better show the advantages of NDN in terms of traffic reduction, we consider a sample simulation for the chain topology shown in Fig. 4(a) and focus on a target BS during the global model download phase. Three clients are connected to the BS and transmit Interests, according to the PCON control mechanism, to retrieve the data packets composing the global model.

Table 4 summarizes the traffic at the targeted BS, when varying the link error rate. Each client has to transmit 16,000 Interest packets to retrieve the global model and the BS is expected to receive 48,000 Interest packets from the three clients (*Expected Interests* in Table 4).

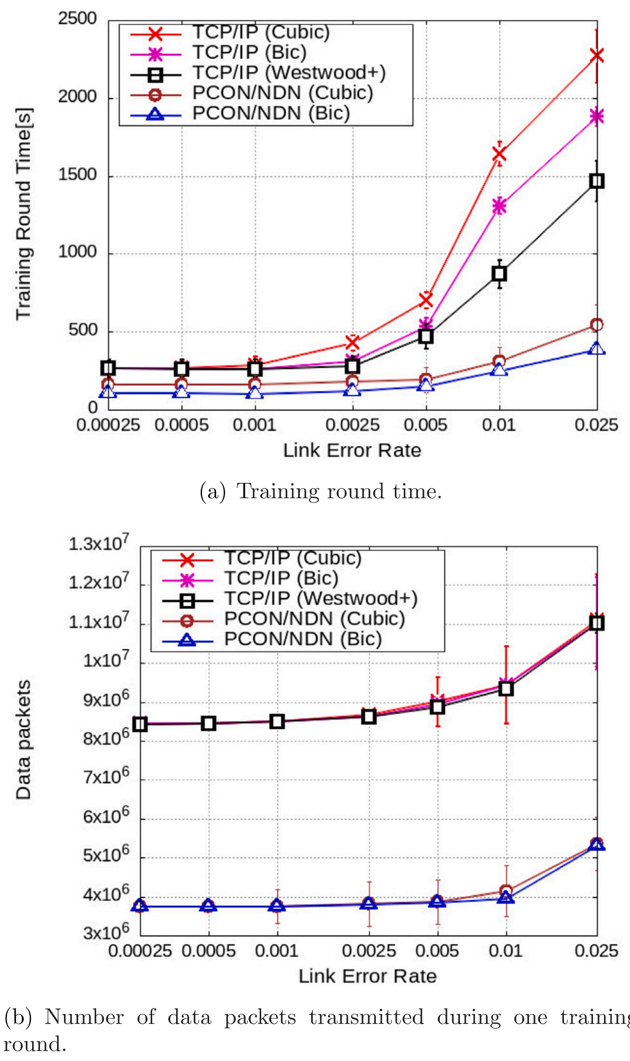


Fig. 7. Metrics when comparing TCP/IP and PCON/NDN approaches in a hierarchical tree topology.

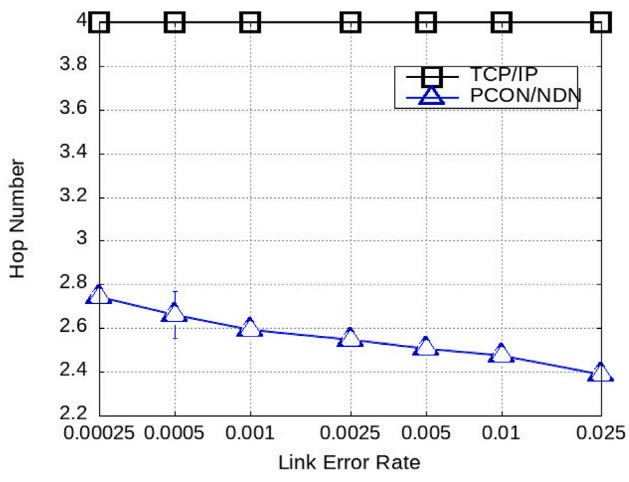


Fig. 8. Number of hops traversed by data packets during the global model download phase in a hierarchical tree topology.

Table 4
Interest and data packets at targeted BS during global model download phase.

Metric	Link error rate = 0.00025	Link error rate = 0.025
Expected Interests	48,000	48,000
Incoming Interests	48,486	62,540
Outgoing Interests	16,000	16,000
Incoming Data	16,000	16,000
Outgoing Data	48,486	62,540
Expected Traffic Reduction [%]	66.6%	66.6%
Actual Traffic Reduction [%]	67%	74.4%

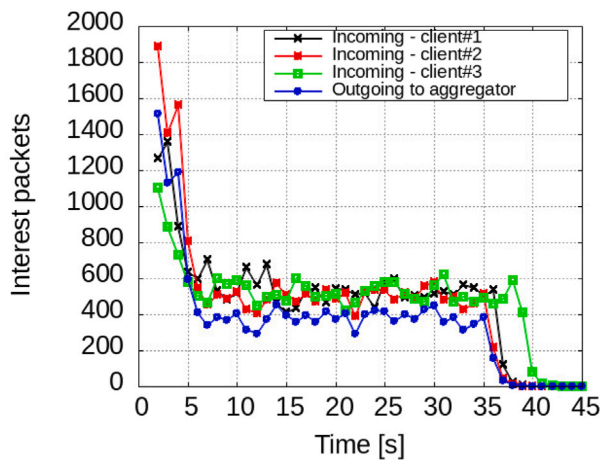


Fig. 9. Received and transmitted Interest packets over time by target BS (Link Error Rate = 0.025).

In practice, however, due to the lossy channels, packets can be lost and unsatisfied Interests are retransmitted by the clients, resulting in a higher actual number of incoming Interest packets at the BS (*Incoming Interests* in Table 4). The number of incoming Interests increases with the error rate because, reasonably, the number of retransmissions is higher. Despite this, thanks to Interest aggregation in the PIT, only 16,000 Interests are actually transmitted from the BS toward the FL server (*Outgoing Interests* in Table 4). Instead, the Data packets sent by the FL server in reply to the aggregated Interests are then sent to each client (*Incoming Data* and *Outgoing Data* respectively, in Table 4).

We measured the traffic reduction ratio, in percentage, computed as the difference between the number of incoming Interests (In_{INT}) and outgoing Interests (Out_{INT}), normalized w.r.t. the number of incoming Interests:

$$Reduction\ [\%] = \frac{In_{INT} - Out_{INT}}{In_{INT}} \cdot 100. \tag{2}$$

Table 4 shows the expected traffic reduction, which should be obtained in ideal channel conditions, i.e., without packet losses. In addition, it can be observed that the actual traffic reduction, experimented by the BS, is even higher than that expected, because of the higher number of incoming Interests.

To further support the previous analysis, Fig. 9 shows the number of incoming and outgoing Interest packets over the simulation time, during the global model download. The BS receives Interests from three distinct clients (see black, red and green curves). However, the number of outgoing Interests is lower than the number of Incoming ones. The results clearly demonstrate that Interest aggregation in the PIT effectively reduces the traffic load on the BS, making NDN a viable solution for optimizing data transmission in FL scenarios.

6.6. Accuracy analysis

So far metrics per round have been measured. Table 5, instead, reports the time of arrival to the desired accuracy measured for the chain topology. In particular, a Top-1 accuracy equal to 81.93% is reached after 148 rounds, when 50 clients are considered. The aforementioned accuracy metric measures the percentage of instances in which the model predicts the correct class as its highest-rank prediction. It can be observed that the NDN-based approach consistently reaches the target accuracy faster than TCP/IP. The time saved by NDN compared to TCP/IP is in the order of 23% for the lowest link error rate settings and increases up to 85% for the worst link error rate considered settings. Indeed, as the error rate increases, TCP is forced to implement end-to-end retransmissions and reduces its sending rate, leading to longer training times. In contrast, NDN maintains efficient delivery thanks to hop-by-hop retransmissions and in-network caching, which mitigate the impact of losses and ensure steady model exchange. Therefore, results confirm the high efficiency of NDN, which is not significantly affected by losses thanks to the enforced countermeasures.

Table 5
Time-to-accuracy [h].

Link error rate	TCP/IP	PCON/NDN
0.00025	4.4543	3.4012
0.0005	4.5202	3.4023
0.001	4.6212	3.4040
0.0025	5.0104	3.4071
0.005	5.3558	3.4127
0.01	8.4725	3.4519
0.025	32.4802	4.6423

7. Conclusion

This paper has provided a comprehensive comparison of host-centric and information-centric communication approaches for FL applications. Through extensive simulations, we demonstrated that ICN solutions, particularly those leveraging NDN, significantly outperform traditional TCP/IP-based approaches in terms of reducing both training round time and the overall traffic transmitted in the network.

The key advantage of NDN lies in its inherent features like in-network caching and multicast support, which efficiently handle packet losses and reduce the need for end-to-end retransmissions. These benefits become especially pronounced in environments with higher error rates and longer distances between clients and the aggregator. In contrast, TCP/IP-based solutions, while effective in stable networks, exhibit performance degradation in error-prone and congested scenarios, even with advanced congestion control mechanisms like Westwood+.

Our findings highlight that NDN offers a promising path forward for cross-device FL scenarios, where communication efficiency is paramount, and numerous clients with heterogeneous capabilities are expected to participate. Future work will focus on optimizing routing strategies to further enhance the benefits of NDN and investigating the potential of client selection mechanisms that can leverage common communication paths for even greater efficiency.

Moreover, the comparison with loss-tolerant approaches will be a further subject matter of investigation.

CRediT authorship contribution statement

Marica Amadeo: Writing – review & editing, Writing – original draft, Validation, Software, Methodology, Investigation, Conceptualization. **Claudia Campolo:** Writing – review & editing, Writing – original draft, Validation, Methodology, Investigation, Conceptualization. **Giuseppe Ruggeri:** Writing – review & editing, Conceptualization. **Antonella Molinaro:** Writing – review & editing, Supervision, Conceptualization.

Acknowledgments

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").

Declaration of competing interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

Data availability

Data will be made available on request.

References

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

[2] Z. Zhao, Y. Mao, Y. Liu, L. Song, Y. Ouyang, X. Chen, W. Ding, Towards efficient communications in federated learning: A contemporary survey, J. Franklin Inst. (2023).

[3] L.U. Khan, W. Saad, Z. Han, E. Hossain, C.S. Hong, Federated learning for internet of things: Recent advances, taxonomy, and open challenges, IEEE Commun. Surv. & Tutorials 23 (3) (2021) 1759–1799.

[4] C. Huang, M. Tang, Q. Ma, J. Huang, X. Liu, Promoting collaborations in cross-silo federated learning: Challenges and opportunities, IEEE Commun. Mag. (2023).

[5] S. Banabilah, M. Aloqaily, E. Alsayed, N. Malik, Y. Jararweh, Federated learning review: Fundamentals, enabling technologies, and future applications, Inf. Process. Manage. 59 (6) (2022) 103061.

[6] C. Zhang, Y. Xie, H. Bai, B. Yu, W. Li, Y. Gao, A survey on federated learning, Knowl.-Based Syst. 216 (2021) 106775.

[7] W.Y.B. Lim, N.C. Luong, D.T. Hoang, Y. Jiao, Y.-C. Liang, Q. Yang, D. Niyato, C. Miao, Federated learning in mobile edge networks: A comprehensive survey, IEEE Commun. Surv. & Tutorials 22 (3) (2020) 2031–2063.

[8] O. Shahid, S. Pouriyeh, R.M. Parizi, Q.Z. Sheng, G. Srivastava, L. Zhao, Communication efficiency in federated learning: Achievements and challenges, 2021, arXiv preprint arXiv:2107.10996.

[9] Z. Jiang, W. Wang, B. Li, Q. Yang, Towards efficient synchronous federated training: A survey on system optimization strategies, IEEE Trans. Big Data (2022).

[10] K. Le, N. Luong-Ha, M. Nguyen-Duc, D. Le-Phuoc, C. Do, K.-S. Wong, Exploring the practicality of federated learning: A survey towards the communication perspective, 2024, arXiv preprint arXiv:2405.20431.

[11] L. Fu, H. Zhang, G. Gao, M. Zhang, X. Liu, Client selection in federated learning: Principles, challenges, and opportunities, IEEE Internet Things J. 10 (24) (2023) 21811–21819.

[12] T. Nishio, R. Yonetani, Client selection for federated learning with heterogeneous resources in mobile edge, in: IEEE ICC, 2019, pp. 1–7.

[13] M.M. Amiri, D. Gündüz, Federated learning over wireless fading channels, IEEE Trans. Wirel. Commun. 19 (5) (2020) 3546–3557.

[14] H. Hellström, V. Fodor, C. Fischione, Over-the-air federated learning with retransmissions, in: 2021 IEEE 22nd International Workshop on Signal Processing Advances in Wireless Communications, SPAWC, IEEE, 2021, pp. 291–295.

[15] P. Pinyoanuntapong, P. Janakaraj, R. Balakrishnan, M. Lee, C. Chen, P. Wang, EdgeML: towards network-accelerated federated learning over wireless edge, Comput. Netw. 219 (2022) 109396.

[16] A. Mahmod, G. Caliciuri, P. Pace, A. Iera, Improving the quality of federated learning processes via software defined networking, in: Proceedings of the 1st International Workshop on Networked AI Systems, 2023, pp. 1–6.

[17] N. Ploch, S. Troia, W. Kellerer, G. Maier, Edge-to-cloud federated learning with resource-aware model aggregation in MEC, in: 2024 IEEE International Conference on Communications Workshops (ICC Workshops), IEEE, 2024, pp. 347–352.

[18] C. Campolo, A. Iera, A. Molinaro, Network for distributed intelligence: A survey and future perspectives, IEEE Access 11 (2023) 52840–52861.

[19] H.K. Gedawy, K.A. Harras, T. Bui, T. Tanveer, Towards context-aware federated learning assessment: A reality check, IEEE Internet Things J. (2023).

[20] G. Mittone, N. Tonci, R. Birke, I. Colonnelli, D. Medić, A. Bartolini, R. Esposito, E. Parisi, F. Beneventi, M. Polato, et al., Experimenting with emerging ARM and RISC-v systems for decentralised machine learning, 2023, arXiv preprint arXiv:2302.07946.

[21] R. Chandrasekaran, K. Ergun, J. Lee, D. Nanjunda, J. Kang, T. Rosing, Fhdnn: Communication efficient and robust federated learning for aiot networks, in: Proceedings of the 59th ACM/IEEE Design Automation Conference, 2022, pp. 37–42.

[22] K. Yu, S. Eum, T. Kurita, Q. Hua, T. Sato, H. Nakazato, T. Asami, V.P. Kafle, Information-centric networking: Research and standardization status, IEEE Access 7 (2019) 126164–126176.

[23] M. Amadeo, C. Campolo, A. Iera, A. Molinaro, G. Ruggeri, Client discovery and data exchange in edge-based federated learning via named data networking, in: IEEE ICC, 2022, pp. 2990–2995.

[24] A. Agiollo, E. Bardhi, M. Conti, N. Dal Fabbro, R. Lazzeretti, Anonymous federated learning via named-data networking, Future Gener. Comput. Syst. 152 (2024) 288–303.

[25] R. Balakrishnan, S. Srikanteswara, N. Himayat, Information centric network protocol for federated learning, 2021, US Patent App. 17/133,314.

[26] S. Mastorakis, A. Afanasyev, I. Moiseenko, L. Zhang, ndnSIM 2.0: A New Version of the NDN Simulator for NS-3, NDN, Technical Report, 2015, NDN-0028.

[27] J. Lee, et al., Data distribution-aware online client selection algorithm for federated learning in heterogeneous networks, IEEE Trans. Veh. Technol. 72 (1) (2022) 1127–1136.

[28] F. Lai, X. Zhu, H.V. Madhyastha, M. Chowdhury, Oort: Efficient federated learning via guided participant selection, in: 15th {USENIX} Symposium on Operating Systems Design and Implementation ({OSDI} 21), 2021, pp. 19–35.

[29] A. Agiollo, P. Bellavista, M. Mendula, A. Omicini, EneA-FL: Energy-aware orchestration for serverless federated learning, Future Gener. Comput. Syst. 154 (2024) 219–234.

[30] A.A. Abdellatif, et al., Communication-efficient hierarchical federated learning for IoT heterogeneous systems with imbalanced data, Future Gener. Comput. Syst. 128 (2022) 406–419.

[31] R. Yu, P. Li, Toward resource-efficient federated learning in mobile edge computing, IEEE Netw. 35 (1) (2021) 148–155.

[32] C. Mwase, Y. Jin, T. Westerlund, H. Tenhunen, Z. Zou, Communication-efficient distributed AI strategies for the IoT edge, Future Gener. Comput. Syst. (2022).

[33] C. Wu, F. Wu, L. Lyu, Y. Huang, X. Xie, Communication-efficient federated learning via knowledge distillation, Nat. Commun. 13 (1) (2022) 1–8.

[34] T. Yang, G. Andrew, H. Eichner, H. Sun, W. Li, N. Kong, D. Ramage, F. Beaufays, Applied federated learning: Improving google keyboard query suggestions, 2018, arXiv preprint arXiv:1812.02903.

[35] W. Wu, et al., SAFA: A semi-asynchronous protocol for fast federated learning with low overhead, IEEE Trans. Comput. 70 (5) (2020) 655–668.

[36] J. Shin, Y. Li, Y. Liu, S.-J. Lee, Fedbalancer: data and pace control for efficient federated learning on heterogeneous clients, in: Proceedings of the 20th Annual International Conference on Mobile Systems, Applications and Services, 2022, pp. 436–449.

[37] J. Lee, H. Ko, S. Pack, Adaptive deadline determination for mobile device selection in federated learning, IEEE Trans. Veh. Technol. 71 (3) (2021) 3367–3371.

[38] M. Yemini, R. Saha, E. Ozfatura, D. Gündüz, A.J. Goldsmith, Semi-decentralized federated learning with collaborative relaying, in: IEEE International Symposium on Information Theory, ISIT, 2022, pp. 1471–1476.

[39] Z. Lin, H. Liu, Y.-J.A. Zhang, Relay-assisted cooperative federated learning, IEEE Trans. Wirel. Commun. 21 (9) (2022) 7148–7164.

[40] M. Fu, et al., Federated learning via unmanned aerial vehicle, IEEE Trans. Wirel. Commun. 23 (4) (2024) 2884–2900.

[41] K. Yang, T. Jiang, Y. Shi, Z. Ding, Federated learning via over-the-air computation, IEEE Trans. Wirel. Commun. 19 (3) (2020) 2022–2035.

[42] Z. Wang, et al., Federated learning via intelligent reflecting surface, IEEE Trans. Wirel. Commun. 21 (2) (2021) 808–822.

[43] G. Cleland, et al., Fedcomm: Understanding communication protocols for edge-based federated learning, in: IEEE/ACM UCC, 2022, pp. 71–81.

[44] Z. Zhang, et al., Is network the bottleneck of distributed training? in: Workshop on Network Meets AI & ML, 2020, pp. 8–13.

[45] C. Campolo, G. Genovese, G. Singh, A. Molinaro, Scalable and interoperable edge-based federated learning in IoT contexts, Comput. Netw. 223 (2023) 109576.

[46] D.J. Beutel, T. Topal, A. Mathur, X. Qiu, J. Fernandez-Marques, Y. Gao, L. Sani, K.H. Li, T. Parcollet, P.P.B. de Gusmão, et al., Flower: A friendly federated learning research framework, 2020, arXiv preprint arXiv:2007.14390.

[47] X. Su, Y. Zhou, L. Cui, J. Liu, On model transmission strategies in federated learning with lossy communications, IEEE Trans. Parallel Distrib. Syst. 34 (4) (2023) 1173–1185.

[48] Z. Chen, L. Shi, X. Liu, X. Ai, S. Liu, Y. Xu, Boosting distributed machine learning training through loss-tolerant transmission protocol, in: 2023 IEEE/ACM 31st International Symposium on Quality of Service (IWQoS), IEEE, 2023, pp. 1–10.

[49] L. Xu, K. Harfoush, I. Rhee, Binary increase congestion control (BIC) for fast long-distance networks, in: IEEE INFOCOM, vol. 4, 2004, pp. 2514–2524.

[50] S. Ha, I. Rhee, L. Xu, CUBIC: a new TCP-friendly high-speed TCP variant, ACM SIGOPS Oper. Syst. Rev. 42 (5) (2008) 64–74.

[51] C. Casetti, M. Gerla, S. Mascolo, M.Y. Sanadidi, R. Wang, TCP westwood: end-to-end congestion control for wired/wireless networks, Wirel. Netw. 8 (2002) 467–479.

[52] L.A. Grieco, S. Mascolo, Performance evaluation and comparison of Westwood+, New Reno, and Vegas TCP congestion control, ACM SIGCOMM Comput. Commun. Rev. 34 (2) (2004) 25–38.
[53] S. Gangadhar, T.A.N. Nguyen, G. Umapathi, J.P. Sterbenz, TCP westwood (+) protocol implementation in ns-3, in: Proceedings of the 6th International ICST Conference on Simulation Tools and Techniques, 2013, pp. 167–175.
[54] C.A. Gomez, X. Wang, A. Shami, Intelligent active queue management using explicit congestion notification, in: 2019 IEEE Global Communications Conference, GLOBECOM, 2019, pp. 1–6.
[55] A. Ford, C. Raiciu, M. Handley, O. Bonaventure, C. Paasch, TCP extensions for multipath operation with multiple addresses draft-ietf-mptcp-rfc6824bis, IETF RFC- 8684 (2016).
[56] S.J. Siddiqi, F. Naeem, S. Khan, K.S. Khan, M. Tariq, Towards AI-enabled traffic management in multipath TCP: A survey, Comput. Commun. 181 (2022) 412–427.
[57] A. Afanasyev, J. Burke, T. Refaei, L. Wang, B. Zhang, L. Zhang, A brief introduction to named data networking, in: MILCOM 2018-2018 IEEE Military Communications Conference, MILCOM, IEEE, 2018, pp. 1–6.
[58] L. Zhang, A. Afanasyev, J. Burke, V. Jacobson, K. Claffy, P. Crowley, C. Papadopoulos, L. Wang, B. Zhang, Named data networking, ACM SIGCOMM Comput. Commun. Rev. 44 (3) (2014) 66–73.
[59] M. Alhowaidi, D. Nadig, B. Hu, B. Ramamurthy, B. Bockelman, Cache management for large data transfers and multipath forwarding strategies in named data networking, Comput. Netw. 199 (2021) 108437.
[60] M. Amadeo, C. Campolo, A. Molinaro, G. Ruggeri, G. Singh, Mitigating the communication straggler effect in federated learning via named data networking, IEEE Commun. Mag. 62 (11) (2024) 92–98.
[61] M. Amadeo, C. Campolo, A. Molinaro, Design and analysis of a transport-level solution for content-centric VANETs, in: IEEE International Conference on Communications Workshops, ICC, 2013, pp. 532–537.
[62] M. Amadeo, A. Molinaro, C. Campolo, M. Sifalakis, C. Tschudin, Transport layer design for named data wireless networking, in: IEEE Conference on Computer Communications Workshops (INFOCOM WKSHPS), 2014, pp. 464–469.
[63] M. Nikzad, K. Jamshidi, A. Bohlooli, A responsibility-based transport control for named data networking, Future Gener. Comput. Syst. 106 (2020) 518–533.
[64] K. Schneider, C. Yi, B. Zhang, L. Zhang, A practical congestion control scheme for named data networking, in: Proceedings of the 3rd ACM Conference on Information-Centric Networking, 2016, pp. 21–30.
[65] S. Mastorakis, A. Afanasyev, L. Zhang, On the evolution of ndnSIM: An open-source simulator for NDN experimentation, ACM SIGCOMM Comput. Commun. Rev. 47 (3) (2017) 19–33.
[66] X. Piao, L. Huang, K. Yuan, J. Yuan, K. Lei, The real implementation of NDN forwarding strategy on android smartphone, in: IEEE 7th Annual Ubiquitous Computing, Electronics & Mobile Communication Conference, UEMCON, 2016, pp. 1–6.
[67] X. Zhang, L. Gong, Y. Xun, X. Piao, K. Leit, Centaur: A evolutionary design of hybrid ndn/ip transport architecture for streaming application, in: IEEE 7th Annual Ubiquitous Computing, Electronics & Mobile Communication Conference, UEMCON, 2016, pp. 1–7.
[68] R. Zhu, T. Li, T. Song, IGate: NDN gateway for tunneling over IP world, in: IEEE International Conference on Computer Communications and Networks, ICCCN, 2021, pp. 1–9.
[69] L.M. Larsen, H.L. Christiansen, S. Ruepp, M.S. Berger, Deployment guidelines for cloud-RAN in future mobile networks, in: IEEE 11th International Conference on Cloud Networking (CloudNet), 2022, pp. 141–149.
[70] A. Rahman, D. Trossen, D. Kutscher, R. Ravindran, RFC 8763 deployment considerations for information-centric networking (ICN), 2020.
[71] G. Carofiglio, M. Gallo, L. Muscariello, D. Perino, Scalable mobile backhauling via information-centric networking, in: IEEE International Workshop on Local and Metropolitan Area Networks, LANMAN, 2015, pp. 1–6.
[72] A. Samanta, Y. Li, DeServE: delay-agnostic service offloading in mobile edge clouds: Poster, in: Proceedings of the Second ACM/IEEE Symposium on Edge Computing, 2017, pp. 1–2.
[73] S. Lange, H.-G. Kim, S.-Y. Jeong, H. Choi, J.-H. Yoo, J.W.-K. Hong, Predicting VNF deployment decisions under dynamically changing network conditions, in: 2019 15th International Conference on Network and Service Management, CNSM, IEEE, 2019, pp. 1–9.
[74] T. Subramanya, D. Harutyunyan, R. Riggio, Machine learning-driven service function chain placement and scaling in MEC-enabled 5G networks, Comput. Netw. 166 (2020) 106980.
[75] I. Tomkos, D. Klonidis, E. Pikasis, S. Theodoridis, Toward the 6G network era: Opportunities and challenges, IT Prof. 22 (1) (2020) 34–38.
[76] B. Mathieu, G. Doyen, W. Mallouli, T. Silverston, O. Bettan, F.-X. Aguessy, T. Cholez, A. Lahmadi, P. Truong, E.M. De Oca, Monitoring and securing new functions deployed in a virtualized networking environment, in: 2015 10th International Conference on Availability, Reliability and Security, IEEE, 2015, pp. 741–748.
[77] J. François, T. Cholez, T. Engel, CCN traffic optimization for IoT, in: Fourth International Conference on the Network of the Future, NOF, 2013, pp. 1–5.
[78] L. Alkwai, A. Belghith, A. Gazdar, S. AlAhmadi, Awareness of user mobility in named data networking for IoT traffic under the push communication mode, J. Netw. Comput. Appl. 213 (2023) 103598.
[79] U. De Silva, A. Lertsinsrubtavee, A. Sathiaseelan, K. Kanchanasut, Named data networking based smart home lighting, in: Proceedings of the 2016 ACM SIGCOMM Conference, 2016, pp. 573–574.
[80] M. Amadeo, C. Campolo, A. Iera, A. Molinaro, Information centric networking in IoT scenarios: The case of a smart home, in: IEEE International Conference on Communications, ICC, 2015, pp. 648–653.
[81] L. Wang, V. Lehman, A.M. Hoque, B. Zhang, Y. Yu, L. Zhang, A secure link state routing protocol for NDN, IEEE Access 6 (2018) 10470–10482.
[82] A. Krizhevsky, et al., Learning multiple layers of features from tiny images, 2009.
[83] A. Howard, et al., Mobilenets: Efficient convolutional neural networks for mobile vision applications, 2017, arXiv preprint arXiv:1704.04861.
[84] S. Zheng, C. Shen, X. Chen, Design and analysis of uplink and downlink communications for federated learning, IEEE J. Sel. Areas Commun. 39 (7) (2020) 2150–2167.
[85] Minimum requirements related to technical performance for IMT-2020 radio interface (s), Rep. ITU- R (2017) M-2410-0.
[86] Y. Cui, K. Cao, T. Wei, Reinforcement learning-based device scheduling for renewable energy-powered federated learning, IEEE Trans. Ind. Informatics 19 (5) (2022) 6264–6274.
[87] P. Wiesner, R. Khalili, D. Grinwald, P. Agrawal, L. Thamsen, O. Kao, Fedzero: Leveraging renewable excess energy in federated learning, in: Proceedings of the 15th ACM International Conference on Future and Sustainable Energy Systems, 2024, pp. 373–385.