Electronic Research Archive

ERA, 32(2): 1333–1364.
DOI: 10.3934/era.2024062
Received: 12 October 2023
Revised: 19 January 2024
Accepted: 23 January 2024
Published: 01 February 2024

http://www.aimspress.com/journal/ERA

Review

A survey on state-of-the-art experimental simulations for privacy-preserving federated learning in intelligent networking

Seyha Ros[1], Prohim Tam[1], Inseok Song[1], Seungwoo Kang[1] and Seokhoon Kim[1,2,*]

[1] Department of Software Convergence, Soonchunhyang University, Asan 31538, Republic of Korea
[2] Department of Computer Software Engineering, Soonchunhyang University, Asan 31538, Republic of Korea

* **Correspondence:** Email: seokhoon@sch.ac.kr.

Abstract: Federated learning (FL) provides a collaborative framework that enables intelligent networking devices to train a shared model without the need to share local data. FL has been applied in communication networks, which offers the dual advantage of preserving user privacy and reducing communication overhead. Networking systems and FL are highly complementary. Networking environments provide critical support for data acquisition, edge computing capabilities, round communication/connectivity, and scalable topologies. In turn, FL can leverage capabilities to achieve learning adaptation, low-latency operation, edge intelligence, personalization, and, notably, privacy preservation. In our review, we gather relevant literature and open-source platforms that point out the feasibility of conducting experiments at the confluence of FL and intelligent networking. Our review is structured around key sections, including the introduction of FL concepts, the background of FL applied in networking, and experimental simulations covering networking for FL and FL for networking. Additionally, we delved into case studies showcasing FL potential in optimizing state-of-the-art network optimization objectives, such as learning performance, quality of service, energy, and cost. We also addressed the challenges and outlined future research directions that provide valuable guidance to researchers and practitioners in this trending field.

Keywords: experimentation; federated learning; intelligent networking; network simulation; privacy preservation

1. Introduction

General data protection regulation (GDPR) has steered into a new era of data protection and privacy awareness [1,2]. As organizations and institutions engage with restricted data privacy requirements, the necessity to find solutions that oblige the regulation and protect sensitive information while still leveraging the potentiality of data-driven learning becomes highly significant. In the era of massive data and internet of things (IoT) heterogeneity, the handling of sensitive data is a complex objective for both businesses and researchers, which expands to a variety of services including healthcare systems, home security, wearable personal, payment systems, etc. [3–6]. The consequences of data breaches and privacy violations include not only legal but also ethical and reputational problems. Due to this circumstance, researchers and practitioners have pushed to explore new paradigms that can reconcile the functionality supports for both data-driven intelligence and robust data protection capability.

Decentralizing data from central servers to individual devices, federated learning (FL) unlocks the power of artificial intelligence (AI) for domains with sensitive data and diverse equipment [7]. It addresses data privacy concerns, enhances security, and improves model generalizability while reducing communication costs and server load [8]. Beyond these general benefits, FL holds enormous potential for network applications. First, its decentralized nature minimizes data transmission to central servers, alleviating network congestion and bandwidth demands [9]. Second, it enables personalized model training on individual devices, leading to contextually relevant models tailored to user behaviors and preferences. Additionally, by leveraging data diversity from various devices and locations, FL generates robust and generalized models that improve the overall accuracy and performance of network applications [10,11]. Ultimately, FL's decentralized approach not only protects data privacy but also optimizes network efficiency, empowers personalization, and boosts model performance in the networking landscape.

For researchers in the networking field, numerous challenges arise as they seek to balance the demands of optimizing network performance and preserving user privacy [12,13]. The exponential growth of data volumes, the increasing heterogeneity of networking taxonomies, and the intense sensitivity surrounding user data all contribute to these challenges. Consequently, researchers are driven to explore novel techniques that lead to a complementary technique, FL, which promises to upgrade the way networking systems are designed and optimized in a distributed and collaborative manner. In 2016, Google researchers released FL as a communication-efficient method of distributed learning between global servers and local participants through iterative global model broadcasting, local training, and model averaging [14]. Figure 1 illustrates the overview of FL that integrates in the networking field, including four main tiers, namely application, network, edge, and cloud. FL (edge) offers a comprehensive set of contributions to networking by introducing a framework where local devices can collectively train a shared model without exposing raw data, particularly sensitive information [15–18]. The framework feature not only enhances privacy preservation but also reduces the communication overhead that often troubles conventional data-sharing or high-volume uploading approaches.

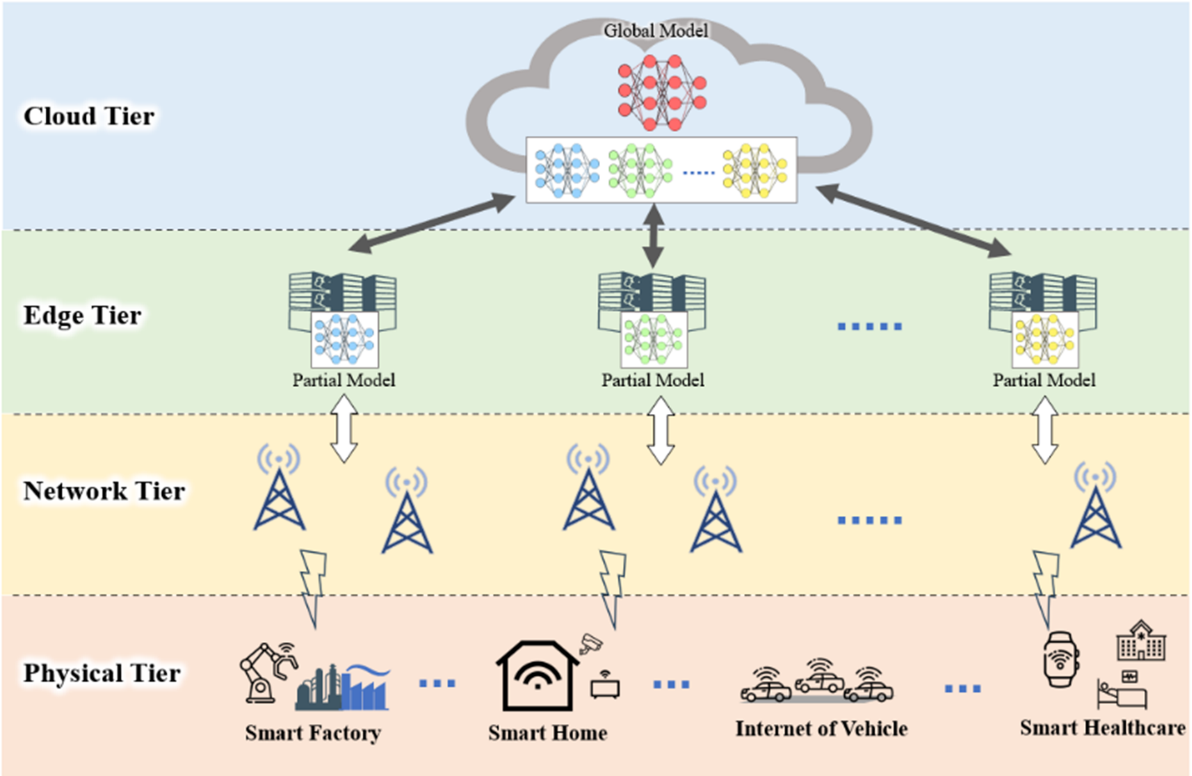

Figure 1. Illustrates the overview of FL that integrates in networking field.

Beyond privacy, FL (edge) also offers the potential for learning adaptation, low-latency operation, edge intelligence, personalization, and cost-effective resource utilization, all of which are significant for intelligent networking systems [19–21]. On the other hand, as researchers begin to explore the relationship between FL and networking, the selection of experimental simulation tools or platforms becomes a critical consideration. Therefore, in this survey, we aim to provide a comprehensive overview of the state-of-the-art experimental simulations for privacy-preserving FL in intelligent networking. Figure 2 presents the paper structure, and Table 1 gives the abbreviations used in this paper. Furthermore, to summarize our contributions, we categorize the main points as follows:

- **Networking for FL:** This section examines network simulation tools to ensure collaboration with FL frameworks. Simultaneously, network platforms can generate a precise dataset to obtain any specific applications. Typically, multiple simulation platforms can provide for several aspects (e.g., network topology acquisition, distributed computing putting capability, multi-round communication, and scalability deployment).
- **FL for networking:** Afterward, we reflect on the abovementioned platforms by reviewing how the FL framework resource can be utilized to complement networking simulation (e.g., data privacy, private preservation, bandwidth-efficient updates, latency reduction, learning robustness in edge intelligence, and converged round communication).
- **Objectives of FL case studies:** As we discussed, the collaboration between multiple simulation tools and the FL framework is eligible to accomplish several case studies in terms of learning performance, QoS, energy efficiency, and cost.
- **Challenges and future directions:** Simulation tools and the FL framework play a crucial role in realizing the full potential of various applications while addressing their inherent challenges in

the critical areas of the research and development of 5G and beyond networks. However, in optimizing communication efficiency between the central server and decentralized clients, especially in constrained network resources, challenges include communication overhead, operational bandwidth cost, and expansion of multi-awareness learning (resources, delays, and energy). Additionally, ensuring the privacy of sensitive data during the FL process remains a paramount concern, prompting investigations into techniques such as secure multi-party computation and differential privacy. The scalability of FL in intelligent networking, mainly when dealing with a vast number of clients and diverse network conditions.

Table 1. List of abbreviations.

Acronym	Description
API	Application programming interface
CAPEX	Capital expenditure
DDQN	Double deep Q-network
DFQL	Deep federated Q-learning
DL	Deep learning
DNN	Deep neural network
DRL	Deep reinforcement learning
DSRA	Device selection and resource allocation
DQL	Deep Q-learning
EC	Edge cloud
E2E	End-to-end
FL	Federated learning
HFL-VNE	Horizontal federated learning-virtual network embedding
IID	Independent-and-identically-distributed
IIoT	Industrial internet of things
IoV	Internet of vehicles
IoT	Internet of things
MEC	Multi-access edge computing
MFL	Multilevel federated learning
ML	Machine learning
MTFL	Multi-tentacle federated learning
NFV	Network functions virtualization
NFVeEC	NFV-enabled EC
QoE	Quality of experience
QoS	Quality of service
RAN	Radio access network
SDN	Software-defined networks
VGAE	Variational graph autoencoder

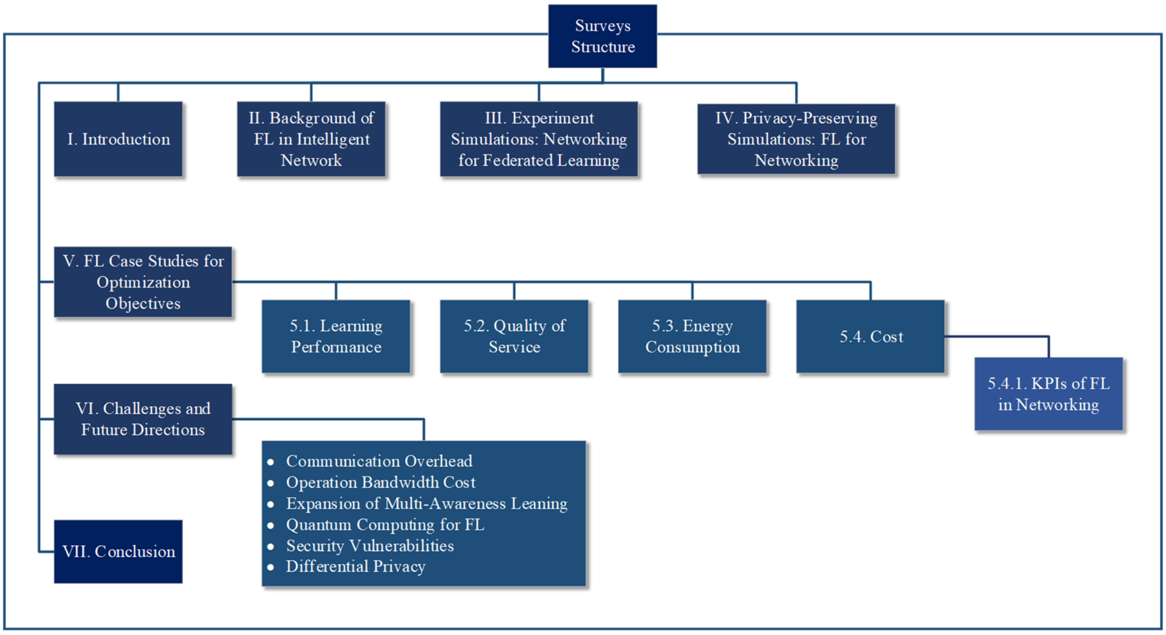

Figure 2. The overall structure of the surveys.

2. Background of federated learning in intelligent networking

FL enables training on decentralized data sources while keeping the data localized. In practical FL scenarios, the active coordinator requires a critical focus on communication efficiency, model aggregation, and security evaluation to ensure its effectiveness. In Table 2, we gather the most relevant surveys analyzing in relation to FL simulation.

Table 2. Describes the summary of existing FL surveys.

Survey paper	IoT networks	Overview of FL	Framework details	Key performance indicators	Simulation tools in networks	Ref.	Year
W. Lim et al.	✓	✓	✓	✘	✘	[22]	2020
D. Nguyen et al.	✓	✓	✓	✘	✘	[23]	2021
R. Gupta et al.	✘	✓	✓	✓	✘	[24]	2022
L. Witt et al.	✘	✓	✓	✘	✘	[25]	2023
H. Chen et al.	✘	✓	✓	✘	✘	[26]	2023
M. Al-Quraan et al.	✘	✓	✓	✘	✘	[27]	2023

We started by listing all the processing steps of conventional FL and pointing out the main functions and literature reviews that are associated with intelligent networking. The execution can be defined as follows:

1) **Initialization**: FL begins with server setup as a global central server that is responsible for coordinating the overall FL process. Then, the coordinator selects the model by creating a model architecture suitable for the target application. Some use cases initially declare the model random or even pre-trained on public data. After deciding on the global model, the FL coordinator is

required to identify the client batch in that round of communication, namely participant selection.

2) **Participant selection**: In mobile networks, participant selection is challenging due to bandwidth limitations and unstable availability or connectivity [28]. Suppose the coordinator sets a policy with random client selection in each round. In that case, the learning performance will return with unreliable accuracy, slow convergence, data imbalance, and unfairness [29], which heavily degrades the practicality in real-world scenarios. The selection approaches tackle the heterogeneity of device taxonomies in modern networks with diverse data type distributions and hardware-specific configurations. Table 3 summarizes proposed approaches in this domain, including targets on resource efficiency, convergence analysis, data heterogeneity handling, power-of-choice strategies, and unstable clients [30–34].

Table 3. Summary of paper contributions on participant selection.

Summary of execution flows	Target objectives	Ref.	Year
The proposed algorithm features two components: 1) An online learning component based on previous resource usage and training results for making fractional decisions, and 2) An internet-based rounding tool that transforms fractional choices into whole number selections.	Prioritization in cumulative utilization of computation and communication resources, while ensuring the convergence of both local and aggregated models.	[30]	2020
The Power-of-Choice selection strategy works as follows: 1) The central server calculates the local losses of all clients, 2) The server sorts the clients in decreasing order of their local losses, and 3) The server selects the top set of clients to participate in the current training round.	Improvement of the convergence speed and accuracy of FL while reducing communication and computation overhead (target applications can be distributed machine learning (ML), mobile edge computing, and IoT).	[31]	2022
"PyramidFL" is a fine-grained client selection method that first determines the utility-based client selection from the global view and then optimizes its utility profiling locally for further client selection.	Designing to speed up the FL training while achieving a higher final model performance, also prioritizing the use of clients with higher statistical and system utility.	[32]	2020
1) Deadline-based aggregation: Aggregate client updates once a fixed deadline is met. 2) Joint optimization: two subproblems on client selection and parameter update. 3) E3CS: an exponential-weight algorithm for exploration and exploitation-based client selection.	Improvement of the training efficiency and final model accuracy in FL within a volatile training context, where clients are prone to failures.	[33]	2022
"Newt" is an enhanced FL approach that includes a new client selection utility and fine-grained control on selection frequency.	Enabling efficient selection in intelligent transportation systems, which has challenges such as data and device heterogeneity and highly dynamic system.	[34]	2022

3) **Model distribution**: After the server selects the clients that will participate in the training round, it distributes the current model to all the selected clients. The clients then download the model onto their local devices and train it on their own data.

4) **Local model training**: Local data is stored on each client device in the FL system, and the clients train on their local data without sending it to the server. However, there are also some challenges associated with using local data, including data heterogeneity, insufficient computation resources, and network connectivity. Due to the problems of local device capability, there are several studies proposing edge-assisted learning for handling FL local tasks, particularly over resource-constrained IoT devices.
5) **Local model updates**: After local training, each client generates the optimal model update for that round, which consists of the model parameters (e.g., weights). Table 4 presents the relevant literature that assists the local model training and updates.

Table 4. Summary of paper contributions on local model training and updates.

Summary of execution flows	Target objectives	Ref.	Year
"FL+HC" clusters clients are grouped based on how closely their local updates match the overall global model. After this grouping, the clusters are trained autonomously and simultaneously using dedicated models.	Improvement of the accuracy of the test set while minimizing the communication rounds needed to achieve convergence in FL, especially when dealing with non-independent and non-identically distributed (non-IID) data.	[35]	2020
1) Bandwidth allocation: involves assigning additional bandwidth to devices that experience poorer channel conditions or possess less robust computational capabilities. 2) Device scheduling: suggests an approach that prioritizes selecting devices with the shortest model updating time until a favorable balance is struck between learning efficiency and round-trip latency.	Maximization of the convergence rate of FL training with respect to time rather than rounds, which can be achieved by minimizing the expected time for FL training to attain certain model accuracy.	[36]	2020
Each client offloads partial data to the edge server for processing, and then, the clients update their local models using the remaining data (Meanwhile, the edge server updates the global model using the aggregated data).	Mitigation of the straggler effect in FL and improving the system efficiency (The method can be used in applications, such as mobile device training, IoT device training, or healthcare data training).	[37]	2021
The optimal offloading strategy is derived by minimizing the FL loss function under the latency constraint and the Q-learning-based offloading strategy is proposed for the imperfect channel state information scenario.	Improvement of training efficiency and mitigating the straggler effect of FL in industrial IoT networks.	[38]	2023
The greedy algorithm is proposed by iteratively assigning communication resources to the edge nodes that consume the least energy for local model training in each round, which proceeds until the communication resource budget is finished.	Training machine learning models for edge-assisted Internet of Agriculture Things applications, where data are both vertically and horizontally partitioned, and resources are limited.	[39]	2022

6) **Model updates aggregation and global update**: Clients securely transmit their model parameter updates (e.g., weights) to the central server or even using edge-assisted methods. The central server aggregates the model updates into a global update. Conventional aggregation methods include federated averaging and secure multi-party computation. The central server integrates the

aggregated global model update into the new-iteration central model. To improve this processing step, there are several studies that tackle different target objectives. Table 5 presents the literature reviews for this phase.

Table 5. Summary of paper contributions on update aggregation and global update.

Summary of execution flows	Target objectives	Ref.	Year
1) Unbiased gradient aggregation: Evaluate the gradients against the global model parameters in the last local epoch, and 2) FedMeta: Perform meta updating on the global model parameters using a small set of data samples indicating the expected target distribution.	Improvement of the convergence speed and accuracy (Both unbiased gradient aggregation and FedMeta can be applied individually or together and can be integrated into existing FL).	[40]	2020
1) Partition the network into groups and local model updates into segments, 2) Apply the aggregation protocol to segments with specific coordination between users, and 3) Aggregate the aggregated segments to obtain the global update.	Enabling secure model aggregation in FL systems, while allowing users to adjust the quantization proportional to their communication resources.	[41]	2022
The method aggregates updates from scheduled devices using an age-aware weighting design, and the weights are assigned to each update based on the freshness of the data, with more recent updates receiving higher weights.	Development of an asynchronous FL framework with periodic aggregation that can support heterogeneous computation capabilities and training data distributions.	[42]	2021
The method uses a feedback mechanism to communicate the estimated global update to each client, which then calculates the relevance of its local update to the estimated global update (If the relevance is lower than a predefined threshold, the client does not update).	Mitigation of communication overhead, which can be applied to a variety of FL applications, such as personalized healthcare, intelligent transportation systems, and distributed machine learning for financial services.	[43]	2019
Participants apply linear transformation to the model update vector, then partially encrypt the transformed vector using multi-input functional encryption.	Design of an efficient secure aggregation scheme, which can protect the security of model updates without losing efficiency.	[44]	2020

Each processing step helps improve the central model performance, deployed after the convergence evaluation and application performance requirements are satisfied. The privacy and security methods (e.g., encryption, secure communication protocols, and access controls) are converged to protect sensitive data and model interactions between entities [45]. Figure 3 illustrates the general view of FL's possible failures in networking. Throughout the overall execution flows, we point out the background of FL and its relationship with networking architecture. There are several challenges to handle, and from a researcher's perspective, complementary simulations of networking and FL are critical to gaining extensive experience before practically deploying their proposed approaches.

This section aims to gather comprehensive information related to FL frameworks and backgrounds studied in networking, especially the execution of FL frameworks in terms of initiate, participatory selection, model distribution, local model training, local mode updates, model update aggregate, and global update.

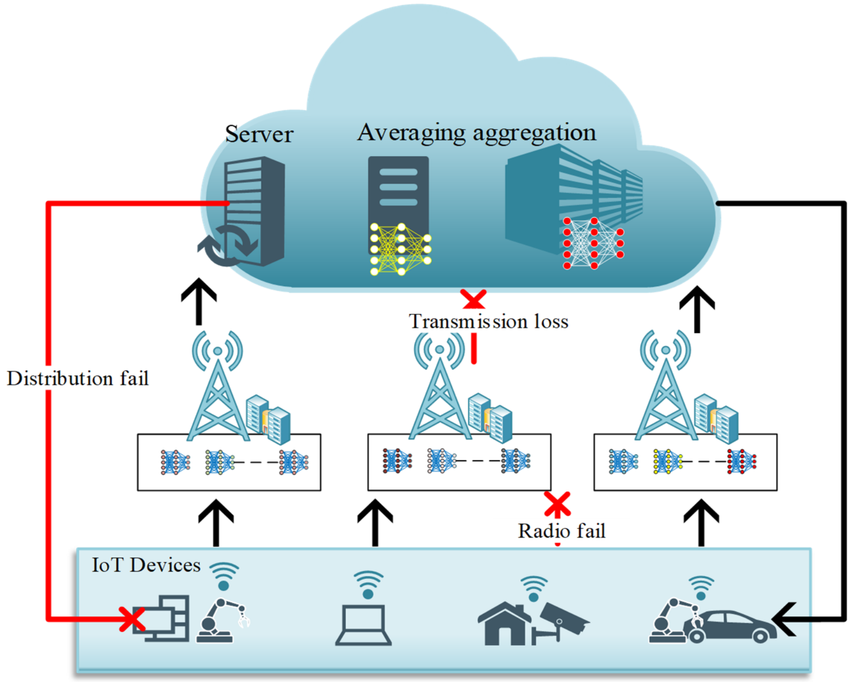

Figure 3. FL processing possible failure in round communications.

3. Experimental simulations: Networking for federated learning

In this section, we primarily concentrate on illustrating the critical role of simulation performance by interacting network components, including user equipment (UE), radio access networks (RAN), core networks (CN), edge cloud (EC), software-defined networks (SDN), multi-access edge computing (MEC), and millimeter wave (mmWave). With several aspects given the intuitionistic abstract of relational network simulation tools, we can tackle representation for network performance in actual implementation and provide the design, optimization, and testing of 5G networking. Additionally, we can provide up-to-date trending networking to customized functions for evaluating and analyzing network performance and results. In this grading, we observe the state of simulation tools to accomplish scale-up, which assisted FL in ensuring the local data deserves security and privacy in many communication and computation aspects [46]. As in Table 6, we gather the trending networking simulation tools, which provide taxonomic and comprehensive information to involve within network protocol, standardization, management, orchestration, and network topology. Simulation tools can be specified depending on the use case and research objectives. The trend of network simulation tools and techniques are listed as follows: network simulation version 3 (NS-3), network simulation version 2 (NS-2), Mininet, Mininet-Wi-Fi, MATLAB, OMNet++, OpenDayLight, Floodlight, Ryu Controller, and OpenStack.

Table 6. Related network simulation tools for networking.

Platform	Summary	Primary focuses	Ref.	Year
NS3 (C++, Python)	A simulator can provide an approach to enhance the comprehensive and flexible platform for computer network simulation. It also provides the ability to analyze and simulate network behaviors and several network standardization protocols, algorithms, scenarios, and topologies. Available: https://www.nsnam.org/	- Customization and extension of its functionality - Supporting comprehensive network environment and network protocols - Execution of the simulation using an XML trace file - Support with OpenAI Gym	[47–49]	2008
NS2 (C++)	Designing to illustrate the behaviors of computer networks, which include wired and wireless communication, routing, and congest control protocol and transport (e.g., routing algorithm). Available: https://www.isi.edu/nsnam/ns/	- Routing - Switching - Extending to support IPv6 - Extending to support cloud computing	[50]	1997
Mininet (Python)	Open-source network emulator that provides to create, configure, and test network topologies, which supports large-scale network infrastructure and the integration with SDN controller using OpenFlow (simplifying the configuration on node, flow entry, and link). Available: http://mininet.org/	- Creation and design for a real-world SDN environment - Interconnecting API and AI - Exportation of network topology into Python script - Being mini-edit to visualize the topology	[51]	2010
Mininet-WiFi (Python)	Designing to evolve rapidly and develop to emulate wireless network controller environment that supports several wireless protocols such as Wi-Fi, ZigBee, and Bluetooth. Available: https://mininet-wifi.github.io	- Development of a new scenario of the network wireless - Measurement of the performance of different Wi-Fi channels - Routing protocol for wireless connection mesh network - Providing a high degree of fidelity - Focusing on the security and scalability of mobile network	[52]	2021
MATLAB (Commercial)	Enabling cooperation with other programming languages and numerical computing environment, which provides more functions for modeling and simulating communication systems, wireless networks, cellular networks, and optical networks (enclosing several tools and libraries). Available: https://www.mathworks.com/help/index.html?s_tid=CRUX_lftnav	- Simulation for network systems and protocols - Simulation for complex networks - Offering customization and development - Specification network function - areas or behaviors	[53]	unde fined

Continued on next page

Platform	Summary	Primary focuses	Ref.	Year
OMNet++ (C++)	Designing a component architecture for models and providing a high degree of scaling to support an extensive network with millions of nodes, which is feasible to obtain a highly accurate result. Available: https://omnetpp.org/ download/models-and-tools	- Support model completeness - Simulation from LANs to WANs - Packet-switched and circuit-switched networks - Support for sensor networks and ad-hoc networks	[54]	1993
OpenDayLight (Java, Python, YAML)	Providing an open platform to focus on customizing and automating networks of network environment, which makes familiar with SDN controller scales up of the network device and network applications. Available: https://www.opendaylight.org	- Combination of multiple service and protocol - Sustenance of various southbound APIs with network devices and protocol - Feasibly support of a wide range of standard networking protocols - Adaptation of NFV and SDN	[55]	2013
Floodlight (Java)	Offering greater flexibility, agility, and programmability in network management. Floodlight makes a rule of network policy and routing logic. Available: https://github.com/floodlight/floodlight	- Support for multiple network protocols - Solution for flexible and scalable approaches	[56]	2014
Ryu-Controller (Python)	Supporting several protocols for managing network devices, such as OpenFlow, Netconf, and OF-config. Available: https://ryusdn.org/index.html	- Creation flow table management - Traffic monitoring - Packet forwarding	[57]	2014
OpenStack	Open source offers a core network controller to accommodate network functions virtualization (NFV), it maintains the feature as a platform for network automation, orchestration, and management resources. Available: https://www.openstack.org	- Network protocols and technologies - A private cloud platform - Enhancement of resources utilization and elastic hypervisors - RESTful API - Tools achieved with automation and integration	[58]	2010

In the 5G perspective, many simulation tools can offer advantages and leverage enhancements to indicate network performance and analysis. Due to differences in network settings, the platforms might gain fluctuation between QoS and QoE. Figure 4 shows the overview of simulations regarding setting scenarios, network environments, and differentiated network tiers; simulation tools can be performed depending on the characteristics of the network.

- NS3 gains interest from researchers in setting the network topologies to compromise the availability of RAN, mmWave, and handover processing. On the other hand, NS-3 has intensified by integrating an Open-AI gym into a collaboration capsule class that allows RL agents.
- MATLAB identifies tools used in many fields that can be simulated and analyzed by various network systems (e.g., communication networks, wireless networks, and network protocols).

Network simulations can be performed on network performance analysis, which consists of analyzing network performance metrics such as latency, packet loss, packet drop ratio, packet size, throughput, and network capacity, by capturing the network routing and using a network protocol to simulate and analyze network routing algorithms such as border gateway protocol (BGP), open shortest path first (OSPF), and routing information protocol (RIP). Within this, MATLAB provides a built-in function and toolbox for simulation with transmission protocol control /internet protocol (TCP/IP), user data gram protocol/ internet protocol (UDP/IP), and Ethernet.

- Mininet is a widely used network emulator and simulator that provides fast simulation speed using OS-level virtualization function and thus has strengths in terms of scalability. It is a simulator that fully supports the OpenFlow protocol and works with various SDN devices. Mininet shows excellent compatibility with various currently released SDN controller open sources. However, the model diversity cannot simulate natural environments and multiple scenarios. The Mininet version includes standard switch and router models (the legacy model) alongside SDN-enabled switches and hosts. Mininet also offers a single link model that can configure detailed parameters like bandwidth, delay, and loss probability. Notably, Mininet lacks its own built-in traffic model and performance analysis capabilities. Instead, it relies on external traffic generation tools such as ping, iperf [59], and D-ITG [60]. Additionally, analyzing performance involves collecting and assessing packets or relying on other external tools. Consequently, there are limitations when using Mininet as a simulation tool for experiments to validate the stability and reliability of networking systems or those requiring a realistic network environment.
- Mininet-WiFi was established early with a wildly supported WiFi modules on the Mininet SDN network emulator and extended the function of the Mininet network by adding virtualized WiFi standard and access point (AP). The modules are based on Linux wireless drivers and 80211hwsim wireless simulation diver. Mininet-WiFi supports the use of spec mode.
- OMNet++ is an open source for academic, non-profit, and industrial purposes. OMNet++ is similar to NS3 and NS2 in the network simulation of both the wired network and wireless network, which holds capability to utilize for several objectives, such as queuing network modeling, mobility, and INET structure. OMNet++ also provides a use case under a graphical interface and limitation of network topology.
- OpenStack is an open source that allows for the building of a simplified platform. It provides flexible customization in implementation, fosters interoperability across deployment, and is easily accessible to meet the requirements of both users and operators (e.g., public and private cloud resource utilization). However, OpenStack can be considered to build a massively scalable operating system with elasticity and horizontal scalability in resources. OpenStack leads the several resource components and orchestration. On the other hand, a web-based application programmable interface (API) ensures that command and control aspects are computation, memory, storage, and networking.
- Floodlight is a controller platform that uses a Java-based OpenFlow controller to support orchestration virtualization and physical infrastructure and manages both OpenFlow and non-OpenFlow networks.
- OpenDayLight (ODL) was introduced by the open network foundation (ONF) in 2013 [61]. ODL controller offers enhanced reliability and clustering capabilities and resolves issues in older controllers. Numerous prominent companies have joined this initiative, aiming to create a resilient and effective system for large-scale industries. ODL significantly impacts SDN's business aspect,

with most controllers choosing it as their foundation.

With several significant FL frameworks and network simulation tools, the primary purpose of conducting experimental simulations in networking for FL is to analyze, evaluate, and validate the performance, behavior, and feasibility of FL algorithms, protocols, and systems within network environments. Those simulations are vital for gaining into how FL operates in diverse network settings, considering factors like different device types, varying communication conditions, network heterogeneity, and privacy concerns. By simulating these scenarios, researchers can fine-tune algorithms, optimize parameters, and understand the implications of deploying FL in practical networking environments without real-world experimentation, accelerating research and development.

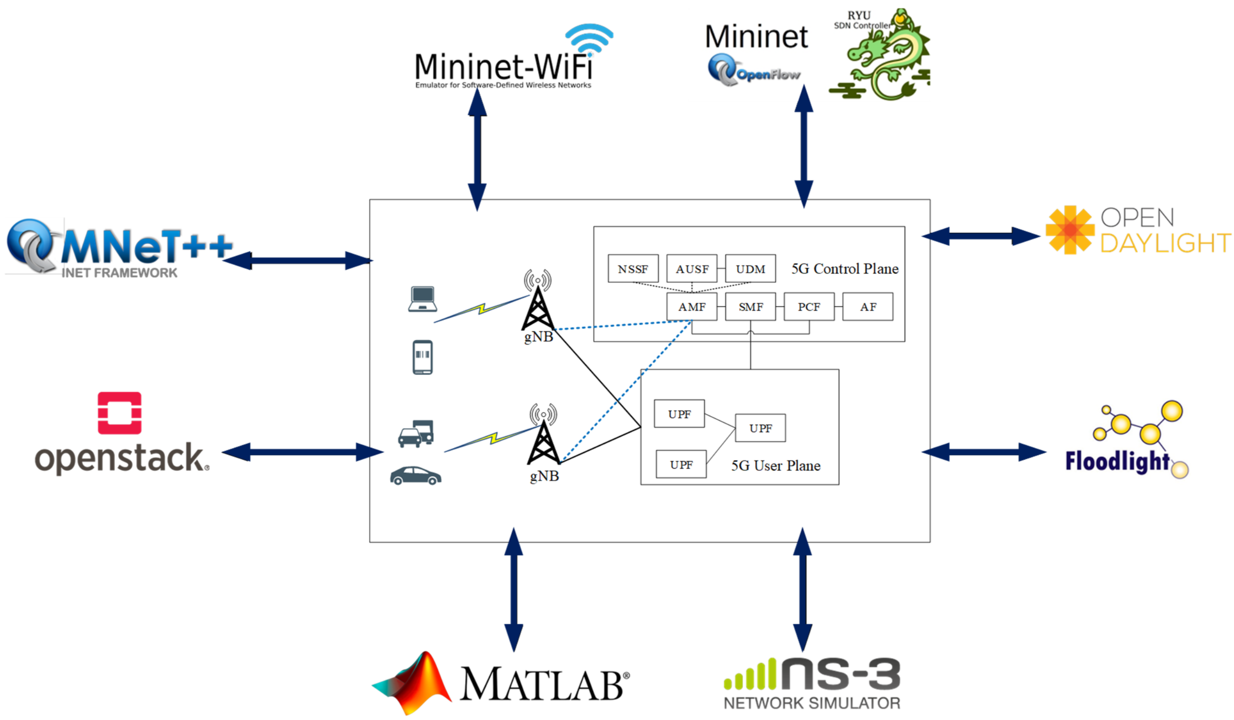

Figure 4. Network simulation tools for collaboration with networking architecture.

4. Privacy-preserving simulations: Federated learning for networking

FL and networking complement each other by leveraging networking's data acquisition, edge and distributed computing capabilities, multi-round communications, and scalability deployment. At the same time, FL reinforces privacy preservation, model updates, latency reduction, learning robustness, and edge intelligence. Table 7 lists well-known platforms for conducting FL experiments. Academia and industry have created numerous platforms to facilitate the widespread adoption of FL, including Google AI, DeepMind, Facebook Research, WeBank+Linux Foundation, IBM, Microsoft, Intel, NVIDIA, etc. All platforms have the capability for the general FL with several dataset supports. We outline the summary specifications and functional focuses as follows:

Table 7. Related platforms to FL experiments.

Platform	Summary	Primary Focuses	Ref.	Year
Flower (Python)	Providing a unified approach to FL, federated evaluation, and federated analytics, which allows users to jointly federate different workloads, ML frameworks, and programming languages. Available: https://flower.dev	- Offering large-scale experiments - Emphasis of heterogeneous participants - Transition to real-world devices - Integration of multiple ML training frameworks - Customizability and extensibility	[62]	2020
FedML (Python)	Prioritization lightweight, cross-platform, and secure FL and federated analytics. Available: https://www.fedml.ai	- Convergence of MLOps tools with decentralized learning - Vertical approaches to industries and applications - Scalability for data silos and rapid large model training	[63]	2020
FATE (Python)	Designing to be industrial-grade with features such as scalability, security, and compliance (supports logistic regression, tree-based algorithms, deep learning, transfer learning, etc.). Available: https://fate.fedai.org	- A wide range of secure computation protocols - FATE CLI, SDK, and dashboard - Design with distributed architecture, message-passing interface, and fault tolerance	[64]	2019
TFF (Python)	Providing a flexible programming model for FL, as well as a variety of tools and libraries to support the development and deployment of FL applications. Available: https://www.tensorflow.org/federated	- High-level interfaces (FL and FC API) including datasets, models, computation builders, client placement type, etc. - Integration with TensorFlow+Keras - Deployment to a variety of runtime environments (mobile devices, edge servers, and cloud platforms)	[65]	2019
PySyft (Python, Docker)	Supporting a variety of techniques, such as differential privacy, secure multi-party computation, and homomorphic encryption. Available: https://openmined.github.io/PySyft/	- Extension to more privacy-preserving techniques - Support for multiple deployment environments - Security and private learning	[66]	2021
FLUTE (Python)	Enabling rapid prototyping and simulation of new FL at scale with cloud integration and multi-GPU support, including novel optimization, privacy, and communications strategies. Available: https://github.com/microsoft/msrflute	- A modular architecture for framework customization - A visualization tool for tracking the progress of FL - A possible integration with AzureML workspaces	[67]	2022

Continued on next page

Platform	Summary	Primary Focuses	Ref.	Year
ns3-fl (C++, Python)	Enhancement of the configuration of network settings and connecting to FL simulator, flsim, while implementation includes client-server ns3 and sync/async FL process communications. Available: https://github.com/eekaireb/ns3-fl	- Emphasis of network settings - A power model of energy consuming while training clients - Enhancement of networking output reliability	[68]	2022
IBM FL (Python)	Enabling distribution of machine learning in enterprise environments, prioritizing privacy, compliance, and data locality, and supporting various learning techniques and topologies. Available: https://ibmfl.res.ibm.com	- Simplification of the adoption with infrastructure, coordination, and compatibility with common libraries and minimizes the learning curve - Deployment across various computing environments and designing custom fusion algorithms - A deep focus on the enterprise environment	[69]	2020
FFL-ERL (Erlang)	Providing Erlang's suitability for FL, comparing its performance in two scenarios: a full Erlang implementation and a hybrid approach using C for numerical computations. Available: https://github.com/gregorulm/fcc_ffl_erl	- Discussion of trade-off between C's high performance and Erlang's superior programmer productivity - Excellence in concurrency and distribution - Development and coordination capabilities	[70]	2018
OpenFL-XAI (Python, Docker)	Emphasis privacy and explainability by focusing on the FL of Fuzzy Rule-Based Systems. Available: https://github.com/Unipisa/OpenFL-XAI	- Intel's OpenFL framework - A solution for accurate, private, and interpretable AI applications - A trustworthy AI systems, privacy preservation, and transparency	[71]	2023
CrypTen (Python)	Enabling privacy-preserving ML with secure multiparty computation with ML-centric features, a tensor library, and robust protocol implementations for real-world use. Available: https://crypten.ai	abstractions for efficient and secure model evaluation - Simplification on secure ML for non-cryptography experts - Integration with PyTorch's familiar API	[72]	2021
Federated Scope (Python, Docker)	Utilization of an event-based structure to grant users significant flexibility in autonomously defining the actions of distinct participants. Available: https://github.com/alibaba/ FederatedScope	- Support plug-in operations and components for enhanced privacy, attack simulation, and auto-tuning - Facilitation of the incorporation of diverse plug-in operations and components to enhance and streamline further development	[73]	2022

Continued on next page

Platform	Summary	Primary Focuses	Ref.	Year
NVIDIA FLARE (Python)	Providing an open-source SDK to ease building workflows with capabilities of scalable packaging, elastic, and lightweight. Available: https://github.com/NVIDIA/NVFlare	- Multiple training and validation workflows - Federated analytics and lifecycle orchestration - Simplification on dashboard for management	[74]	2020
EasyFL (Python)	Targeting beginners with limited prior knowledge, while offering low-code platform, API design, and enhancing the deployment efficiency. Available: https://github.com/EasyFL-AI/EasyFL	- Training outputs tracker - Simulation with statistical and system heterogeneity - Design on plug-in for training flow abstraction	[75]	2022
LEAF (Python)	Providing an adaptable benchmarking system designed for evaluating learning in federated settings with a collection of openly available federated datasets. Available: https://github.com/TalwalkarLab/leaf	- Datasets: FEMNIST (Image classification), Sentiment140 (Sentiment analysis), Shakespeare (Next-character prediction), Synthetic (Classification), and Reddit (Next-word prediction) - Emphasis on learning in federated settings	[76]	2018
PaddleFL (Python, C++, Kubernetes)	Leveraging distributed training and Kubernetes-based job scheduling to offer scalable deployment, also easy replication. Available: https://github.com/ PaddlePaddle/PaddleFL	- Simplification of the deployment of FL systems on large-scale distributed clusters - A flexible framework with components for defining tasks, designing ML models, and handling distributed training configurations - Detail with run times on server, worker, and scheduler	[77]	2020

Figure 5 illustrates the overview of relations between FL for networking and networking for FL. These platforms provide comprehensive support for FL by offering various tools, libraries, and features tailored to different use cases and requirements. Flower stands out for its unified approach, emphasizing large-scale experiments, heterogeneous participant support, and multiple ML training frameworks. FedML prioritizes lightweight and secure FL, converging MLOps tools, and offering vertical approaches for various industries. FATE focuses on industrial-grade FL, supporting various algorithms, secure computation protocols, and a distributed architecture. TFF provides a flexible programming model, high-level interfaces, and easy deployment to diverse runtime environments. PySyft specializes in privacy-preserving techniques, offering support for multiple deployment environments. FLUTE enables rapid prototyping and simulation of FL at scale with modular architecture and cloud integration. ns3-fl enhances network settings and communications for FL simulation, while IBM FL focuses on enterprise environments, minimizing the learning curve and supporting custom fusion algorithms. FFL-ERL leverages Erlang concurrency and distribution capabilities for FL, OpenFL-XAI emphasizes privacy and explainability, and CrypTen bridges secure multiparty computation and ML. FederatedScope supports plug-in operations and components, NVIDIA FLARE simplifies workflows, EasyFL targets beginners with low-code and efficient deployment, and LEAF offers federated benchmarking with openly available datasets. Finally, PaddleFL simplifies deployment on large-scale clusters, providing flexibility for defining tasks and handling distributed training configurations, making these platforms valuable assets for the FL community. The lists of tools in Tables 6 and 7 complement each other in Table 8, which surveys articles about FL framework and simulation tools.

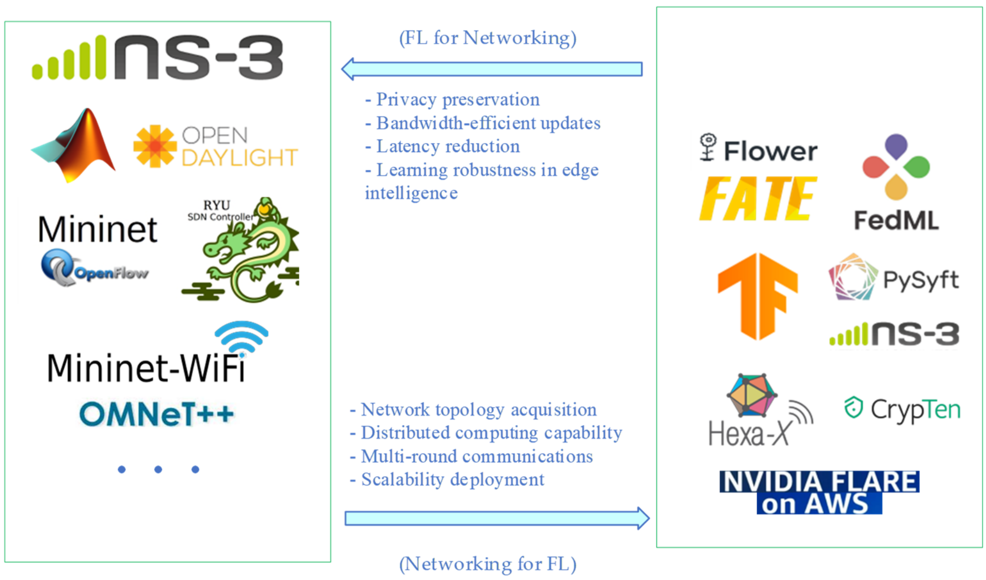

Figure 5. Federated learning for network simulation.

Table 8. Surveys on utilized FL framework and network simulation tools.

Paper	Performance matric	Aggregation approach	FL framework	Network simulation	Data acquisition	Ref.	Year
L. Sami et al.	Throughput	FedAvg	Flower, FedScale, Flue	Unspecified	Google speech, OpenImage, Shakespeare	[78]	2023
P. Tam et al.	Drop ratio, delivery ratio, delay, accuracy	FedAvg	TensorFlow Federated	Mininet, Ryu	MNIST and topology simulation	[79]	2021
V. Balasubramanian et al.	Cache hit ratio, average delay	FedCo	PyTorch	Mininet	Simulation	[80]	2021
R. Uddin et al.	Precision, recall, F1-score, accuracy	AWS OpenGrid	OpenMined	Mininet, Ryu	STIN SAT20, simulation (SDN)	[81]	2023
V. Balasubramanian et al.	MNO revenue, cache hit ratio, average access latency, percentage error in placement	FedCo	Python	Mininet-Wifi	Simulation	[82]	2021

5. FL case studies for optimization objectives

In this section, we brought up five primary sector domains that are discussed about learning performance, QoS, energy consumption, and cost that recent studies have proposed using FL-based approaches for networking as Figure 6 and Table 9.

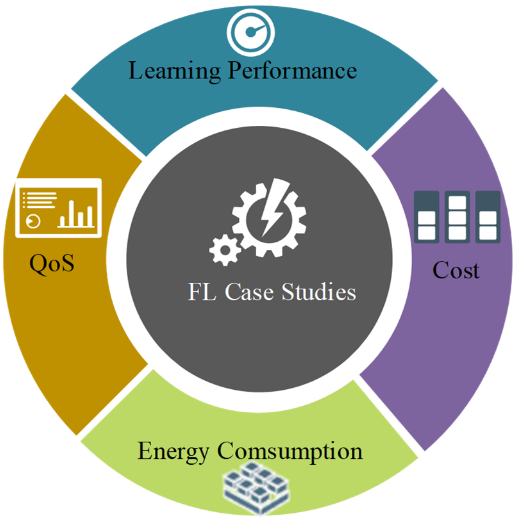

Figure 6. FL case studies for optimization.

5.1. Learning performance

FL with networking is an extension of collaborative ML approaches that allows multiple decentralized devices or servers to collaborate to train shared models while keeping local and private data. FL has been increased to coordinate with learning performance in terms of computational capacity, amount of memory, and communication resources, which contribute to the learning process of privacy preservation for sensitive information.

In [83], authors proposed SDN support to the FL. Implementing the FL framework (FL client and FL server) is supported at the application level and network layer (SDN controller). SDN is handled with data empirically in the FL process, and this paper focuses on the less-explored issue of using FL for high-sensitive application applications. Hence, edge devices have experienced delays due to communication overload. Lastly, SDN maintains efficiency FL even in meeting heavy conditions. SDN-assisted FL can significantly reduce processing time with minimal signaling overhead to the form of the controller. Moreover, the authors aimed to enhance the security and trustworthiness of FL in the software-defined industrial internet of things (SD-IIoT) [84,85]. Implementing multi-tentacle FL (MTFL) frameworks assists in solutions to the growing prevalence of poisoning attacks. MTFL groups participants with similar training data and mode parameters into a "tentacle group". Along with impacting adaptive poisoning data, a stochastic tentacle data exchanging (STDE) protocol was utilized. The protocol involved adding Gaussian noises to exchange data, which differs from traditional defense mechanisms in that all exchanged data will be processed using differential privacy technology to safeguard tentacle privacy.

5.2. Quality of service

QoS is the major evaluation metrics for both aspects, namely E2E networking and FL multi-round communications. Each metric is essential for guaranteeing the delivery of services, especially in mission-critical applications. Several aspects of FL frameworks are to ensure that it can more effectively and precisely match the QoS requirements of the application. QoS indicates important things within the premises that as massive data gathering and data transmission [86]. Edge caching based on the fog computing network is considered a provisioning potential solution to tackle the latency and further content fetch delay and minimize the QoS of the end-to-end (E2E) delay. To prevent network congestion, bottlenecks, and the risk of data user privacy leakage, FL is ultimately used to enable E2E-assisted fog computing networks and edge virtualization [87–89]. In a novel multi-center FL framework for QoS prediction which utilized the central server from the cloud to the edge and trains global models in edge regions [90]. This framework employs two gradient aggregation strategies: 1) internal aggregation for regional users and 2) external aggregation among edge servers with cloud server assistance.

5.3. Energy consumption

When referring to FL for network environment, energy and power consumption are important factors as recently demonstrated by the many reviewed studies. Because FL distributes the training process across devices on the network, interruptions such as power supply issues, network disconnections, and other disruptions can lead to a device pausing or discontinuing the learning process after a certain number of epochs. Moreover, when discussing the FL approach, it is essential to acknowledge the necessity of constructing ML models efficiently. These models should be capable of acquiring knowledge from historical IoT sensor data, which is employed to enhance energy efficiency and reduce costs [91]. Therefore, the significance of optimizing the energy consumption objective becomes more pronounced when taking the entire FL-IoT context. [92] presents the problem as one that centered on reducing both the delay and energy consumption of IoT devices within FL systems. The authors considered various factors, including the decision of whether to offload tasks to edge or cloud resources, the allocation of computation resources, local processing, and transmission power. Their primary objective is to minimize the long-term delay and energy usage. This problem was a multi-agent DRL challenge, which they tackled using a double-deep Q-learning (DQL) network through a DQL agent. To address the decision of whether to offload a task for processing. In the subsequent phase, they focused on allocating computation and communication resources [93]. Moreover, they employed the FedRL approach, where each IoT device trained its own DQL models, shared these models with a centralized controller, and updated the models in a central aggregation unit [94]. QoS for tasks and task queue length, exhibited fluctuations, selecting an efficient update frequency for the target network to stabilize the environment, leading to improved solutions for the agent. It is important to mention that the authors did not discuss the results related to cost minimization in their study.

Table 9. Surveys on utilized simulation tools and the FL framework.

Categorize	Methodology	Significance	Performance matric	Simulation/framework	Ref.	Year
Learning performance		Edge client devices.	Accuracy and loss.	GNS3, OpenDayLight, NETCONF and RESTCONF/Flower, FedAvg	[83]	2023
	TD-EPAD algorithm	Trustiness of training data in SD-IIoT.	Accuracy.	Mininet, Floodlight	[84]	2023
Quality of service	Supporting vector machine	Handling on allocating the network resource and controlling massive communication in backbone.	Delay, jitter, packet drop ratio, packet delivery ratio, throughput.	NS-3	[87]	2021
	CUPE algorithm	Minimization of the content fetch delay for latency- sensitive of IoT.	Cach hit rate, content fetch delay.	Unspecified	[88]	2023
	DDQN	Overcoming on the minimization of energy consumption, completion time, and round communication between local participant and node selection.	Packet drops ratio, throughput, overall accuracy, delay, and packet delivery ratio.	NS-3, Mininet, mini-nfv, OASIS TOSCA	[89]	2022
	MultiFed, a multi-center FL framework	Acceleration for better convergence, heterogeneity handling, and improvement on scalability for privacy preservation in cloud-edge collaboration.	Total communication rounds, total communication delay, and total communication volume.	Unspecified	[90]	2023
Energy consumption	MIBLP, DDQN, FedRL algorithm	Minimization of the long-term delay and energy consumption of an IoT device.	Cost per user.	Unspecified	[92]	2021
Cost	Fed-average algorithm	Approvement on client selection and optimizing model aggregation.	Averaging training accuracy, energy consumption of unit loss delay.	Unspecified	[95]	2022

5.4. Cost

When considering the goal of optimizing network costs, the primary focus is on two components: operational expenses (OPEX) associated with capital expenditure (CAPEX). For instance, in the context of FL implementations, communication often acts as a significant bottleneck, and thus, reducing communication costs involves optimizing connections [95]. Within this section, one can observe a variety of approaches to representing costs in the FL-IoT environment. Different research studies have been adopted for various metrics to achieve cost optimization. As a result, we encounter different terminologies used by different authors in this section. To enhance clarity, we have provided the specific cost metric for each study. [96] considers the battery-constrained federated edge learning problem for their operating CPU frequency for both prolonging the battery life and avoiding the untimely dropping from FL training. The primary objective was to reduce the overall system which was determined by both latency and energy consumption. To achieve this, the authors devised a resource allocation scheme to adjust the CPU frequency of devices and allocate wireless bandwidth. They introduced a DDPG-based allocation strategy to address this issue. Under this approach, clients were required to report their available resources and the server had to estimate the channel parameters for communication between users and devices. Additionally, the base station had to inform participants about the current CPU frequency and available wireless bandwidth. Consequently, the agent received varying rewards to meet both latency and energy consumption requirements. The proposed DDPG strategy outperformed E-DDPG (DDPG strategy with even bandwidth allocation) in terms of system cost. Mainly, it effectively utilized wireless bandwidth resources during the learning process, resulting in improved system performance. Moreover, DDPG performed better than alternative methods across various bandwidth values, as it efficiently leveraged system communication and computational resources.

5.4.1. Key performance indicators of FL in networking

Key performance indicators (KPIs) in FL in networking aim to measure the effectiveness and overall performance of collaborative model training of FL systems across decentralized devices while preserving user privacy. Here are the following key aspects:

- **Speed and convergence:** Introduces a novel framework that optimizes FL by employing multiple edge servers. This framework likely includes mechanisms for efficient communication, aggregation of model updates, and strategies to minimize latency during the learning process [97]. It also discusses empirical results and performance evaluations of FedMes, showcasing improvements in FL speed, convergence rates, and overall efficiency compared to traditional FL setups. [98] The primary focus is on improving the speed and convergence of FL algorithms, specifically by leveraging the momentum gradient descent (MGD) technique. MGD can be a valuable technique for accelerating FL, leading to faster model training, and potentially improving resource efficiency. The proposed adaptive FedMGD further enhances the performance by adapting to device heterogeneity.
- **Delay:** The majority of techniques must be solved to accelerate the model updates and parameters between the central server and participating devices. [99] The authors propose a dynamic sampling and adaptive resource allocation (DSARA) framework to minimize service delays in mobile FL. The algorithm incorporates the latest local model updates and resource constraints to

select the optimal device subset for each round. It offers a valuable solution for enabling efficient and delay-aware FL on mobile devices, paving the way for practical applications in mobile edge computing and distributed ML.

- **Communication round:** The backbone of information exchange in FL, enabling collaborative model training while preserving data privacy on participating devices. 1) infrequent rounds with compressed updates achieved decent accuracy while minimizing bandwidth usage, 2) strategic selection of participating devices based on data relevance further improved efficiency, and 3) federated averaging with weights assigned based on update quality accelerated convergence towards the optimal model. Furthermore, it demonstrates the importance of optimizing communication rounds for resource-constrained networks in real-world FL applications. Moreover, [100] aims to decrease the size of models produced by both the server and clients while adjusting the FL procedure. Initially, the server creates a smaller sub-model with fewer parameters using federated dropout. Subsequently, the resulting model undergoes lossy compression on the server's end and transmits to the clients. The clients then decompress the model to begin training. After training, the updates are compressed and sent back to the server, which decompresses and combines the final model. The communication bottleneck in the federated averaging method is due to limited bandwidth, causing delays for clients in uploading their updates.
- **Resource utilization:** In FL-based IoT environments, we observe many heterogeneous IoT devices that are in nature and process limited resources. Hence, FL trains operated through efficient client selection and optimized resource utilization. In [101], it tackled the hurdles of unreliable wireless communication in FL. They built a framework that joins learning tasks with wireless communication on multiple devices. Recognizing issues like packet errors and limited bandwidth, they devised an optimization plan to minimize training errors while allocating resources and wisely choosing participating users. The framework accounts for how wireless channels affect learning, leading to a precise formula for predicting how well the model will train. This breakthrough could pave the way for reliable and efficient FL in resource-constrained settings like wireless networks.

This section focuses on utilizing FL case studies regarding learning performance, QoS, energy consumption, and cost. Meanwhile, KPIs are primarily impacted by FL to show the effectiveness of its ability to adapt to speed and convergence, delay, communication round, and resource utilization.

6. Challenges and future directions

This section presents the main research challenges and future directions in this recent review within several issues to tackle for supporting FL with network simulation tools. However, this field also faces several challenges and has numerous future directions to explore, as follows:

- **Communication overhead**: To prevent communication overhead during communication between simulation tools and other tools for collaborating interactions, which is a critical phase to consider accurately in network environments of heterogeneous IoT. The primary purpose of maintaining communication and synchronization among various network components is the interface, nodes, or entities involved. For instance, where a client faces limited bandwidth, effective communication with the FL server during model training becomes challenging. Similarly, clients with insufficient processing capabilities find it impractical to execute assigned local computational tasks [102–105]. Additionally, when dealing with extensive data across the network, the resulting large model size poses difficulties for resource-constrained clients. Efficient training in such large data networks necessitates compressing client models, minimizing the burden on clients with constrained resources. If many FL clients grapple with resource limitations, the FL process demands increased server-client interactions to achieve target convergence. However, clients may find bearing the associated high communication costs prohibitive.

- **Operational bandwidth cost:** Offering the operational bandwidth cost is one of the concern cases in networking operations and a critical consideration as FL relies on the exchange of data and model between participants and global server. The following factors can involve operational bandwidth costs in FL: 1) exchanging data for local collaboration, 2) the frequency of participant-server round communications, 3) the distance/mobility between participants, 4) networks with lower bandwidth capacity [106] (cellular networks) will have higher bandwidth costs than networks with higher bandwidth capacity [107], and 5) the number of simulators/platforms involved in the deployment. Therefore, utilizing a (edge) cloud-based FL platform can optimize bandwidth cost by optimizing data transfer, edge aggregation, and caching capabilities.
- **Expansion of multi-awareness learning (MAL):** FL framework enables the ML model to learn from distributed data while preserving privacy and reducing communication costs. Network simulation offers exciting opportunities for enhancing network performance through innovative methods. For instance, MAL models can dynamically fine-tune routing algorithms, congestion control protocols, and power management strategies, improving network throughput, decreasing latency, and energy conservation. One approach for extending MAL to network simulation is to adopt a reinforcement learning methodology. In this method, the MAL model is trained to interact with a simulated network environment, learning to take actions that maximize a predefined reward. Reward functions can be tailored to represent specific network performance objectives, such as achieving higher throughput, lower latency, or reduced energy consumption. Another approach for integrating MAL into network simulation involves using supervised learning. The MAL model was trained using a historical network data dataset annotated with desired network performance metrics. The expansion of MAL into network simulation presents a promising avenue of research with the potential to transform how networks are conceived and administered. MAL can create more efficient, dependable, and sustainable network infrastructures by empowering ML models to glean insights from distributed data and adapt to evolving network conditions. The specification terms of how MAL could optimize network performance are routing optimization, congestion control, and power management. The limitation emerges when a network characteristic relies on the models of its components. These models tend to oversimplify the real-world scenario and the performance of real-world networks is impacted by the variability in conditions, including traffic loads, hardware failures, and software bugs. These factors can be challenging to model accurately in a simulator.
- **Quantum computing for FL:** Quantum computing presents the opportunity for substantial benefits compared to classical machine learning algorithms [108]. By harnessing quantum superposition and entanglement, quantum algorithms can conduct computations in parallel, potentially leading to faster processing and heightened efficiency. Quantum ML (QML) algorithms exhibit superior effectiveness in managing extensive datasets and intricate patterns, enhancing learning capabilities. Moreover, quantum algorithms can extract information from quantum state transformations and interference, resulting in more accurate predictions. While the realization of

quantum advantage in QFL is an ongoing area of research and development, the distinctive features of quantum computing show potential for addressing complex computational tasks and introducing novel possibilities in data analysis and pattern recognition. It challenges the following tantalizing options: 1) collaborative learning across a vast network, 2) data privacy, 3) communication, 4) gradient leakage, 5) compromised client, and 6) compromised server.

- **Security vulnerabilities:** The distributed training method in FL provides an avenue for engaging a substantial client base, potentially extending to millions of clients. Within the FL framework, clients are not inherently trustworthy and each client may possess varying degrees of malicious or adversarial capabilities. [109] The server employs a random selection process for clients, allowing them to participate in FL training iterations. Identifying malicious clients in a training session becomes challenging when dealing with thousands or even millions of clients. Consequently, malicious clients may exploit the system to gain access to and learn about the privacy of other clients participating in the same iteration.
- **Differential privacy:** DP may lead to exposing sensitive data information. FL can preserve privacy across various applications, including social media, innovative city applications, healthcare [110], and traffic management. In these applications, sensitive data is stored locally on mobile or edge devices [111]. Only the model parameters derived from these devices are transmitted to the global FL to train the overarching machine-learning model. The subsequent discussion delves into recent studies and efforts to utilize FL to preserve privacy in these contexts.

At the end of this section, FL demonstrates its increasingly significant role in IoT networks and applications. We also raise several critical research challenges and current issue trends to be considered for further FL-networking system implementation. We list several challenges concerning FL with network simulation, such as communication overhead, operation bandwidth cost, expansion of multi-awareness learning, security vulnerabilities, and differential privacy.

7. Conclusions

In this paper, we presented a survey of experimental simulations for privacy-preserving FL in intelligent networking. We examined the potential relationship between the network simulation tool and the FL framework. An introduction with a motivational statement from networking and FL is gathered comprehensive terms in novelty concepts and technologies recently. Then, we described the preliminary studies for FL frameworks that can be leveraged to achieve learning approaches with intelligence networks. Afterward, we provided networking environments that provide critical support for data acquisition, edge computing capabilities, round communication, connectivity, and scalable topologies. Moreover, FL can leverage capabilities to achieve learning adaptation, low-latency operation, edge intelligence, personalization, and privacy preservation. Additionally, we brought up case studies of FL potential in terms of learning performance, QoS, energy consumption, and cost. Finally, we discussed potential challenges and future directions that could provide valuable ways for researchers in the recently trending field development to enhance the application in practical, real-world systems.

Use of AI tools declaration

The authors declare they have not used Artificial Intelligence (AI) tools in the creation of this article.

Acknowledgments

This work was supported by Institute of Information & communications Technology Planning & Evaluation (IITP) grant funded by the Korea government (MSIT) (No. RS-2022-00167197, Development of Intelligent 5G/6G Infrastructure Technology for The Smart City), in part by the National Research Foundation of Korea (NRF), Ministry of Education, through Basic Science Research Program under Grant NRF-2020R1I1A3066543, in part by BK21 FOUR (Fostering Outstanding Universities for Research) under Grant 5199990914048, and in part by the Soonchunhyang University Research Fund.

Conflict of interest

The authors declare there is no conflict of interest.

References

1. N. Gruschka, V. Mavroeidis, K. Vishi, M. Jensen, Privacy issues and data protection in big data: A case study analysis under GDPR, in *IEEE International Conference on Big Data (Big Data)*, (2018), 5027–5033. https://doi.org/10.1109/BigData.2018.8622621
2. M. Rhahla, T. Abdellatif, R. Attia, W. Berrayana, A GDPR controller for IoT systems: Application to e-Health, in *IEEE 28th International Conference on Enabling Technologies: Infrastructure for Collaborative Enterprises (WETICE)*, (2019), 170–173. https://doi.org/10.1109/wetice.2019.00044
3. X. Yu, Y. Yang, W. Wang, Y. Zhang, Whether the sensitive information statement of the IoT privacy policy is consistent with the actual behavior, in *Annual IEEE/IFIP International Conference on Dependable Systems and Networks Workshops (DSN-W)*, (2021), 85–92. https://doi.org/10.1109/dsn-w52860.2021.00025
4. P. Liu, S. Ji, L. Fu, K. Lu, X, Zhang, J. Qin, et al., How IoT re-using threatens your sensitive data: Exploring the user-data disposal in used IoT devices, in *IEEE Symposium on Security and Privacy (SP)*, (2023), 3365–3381. https://doi.org/10.1109/sp46215.2023.10179294
5. C. Thirumalai, H. S. Kar, Memory efficient multi key (MEMK) generation scheme for secure transportation of sensitive data over cloud and IoT devices, in *Innovations in Power and Advanced Computing Technologies (i-PACT)*, (2017), 1–6. https://doi.org/10.1109/ipact.2017.8244948
6. W. Xu, T. Xiao, J. Zhang, W. Liang, Z. Xu, X. Liu, et al., Minimizing the deployment cost of UAVs for delay-sensitive data collection in IoT networks, *IEEE/ACM Trans. Networking*, **30** (2022), 812–825. https://doi.org/10.1109/tnet.2021.3123606
7. R. Parasnis, S. Hosseinalipour, Y. W. Chu, M. Chiang, C. G. Brinton, Connectivity-aware semi-decentralized federated learning over time-varying D2D networks, in *ACM on Mobile Computing and Communications (MobileCom)*, (2023), 31–40. https://doi.org/10.1145/3565287.3610278

8. P. Qi, D. Chiaro, A. Guzzo, M. Ianni, G. Fortino, F. Piccialli, Model aggregation techniques in federated learning: A comprehensive survey, *Future Gener. Comput. Syst.*, **150** (2024), 272–293. https://doi.org/10.1016/j.future.2023.09.008
9. M. Chahoud, S. Otoum, A. Mourad, On the feasibility of federated learning towards on-demand client deployment at the edge, *Inf. Process. Manage.*, **60** (2023), 103150. https://doi.org/10.1016/j.ipm.2022.103150
10. A. Rahan, K. Hasan, D. Kundu, Md. J. Islam, T. Debnath, S. S. Band, et al., On the ICN-IoT with federated learning integration of communication: Concepts, security-privacy issues, applications, and future perspectives, *Future Gener. Comput. Syst.*, **138** (2023), 61–88. https://doi.org/10.1016/j.future.2022.08.004
11. G. Lan, X. Y. Liu, Y. Zhang, X. Wang, Communication-efficient federated learning for resource-constrained edge devices, *IEEE Trans. Mach. Learn. Commun. Networking*, **1** (2023), 210–224. https://doi.org/10.1109/TMLCN.2023.3309773
12. C. Zhang, J. Sun, X. Zhu, Y. Fang, Privacy and security for online social networks: Challenges and opportunities, *IEEE Network*, **24** (2010), 13–18. https://doi.org/10.1109/mnet.2010.5510913
13. K. Yang, K. Zhang, J. Ren, X. Shen, Security and privacy in mobile crowdsourcing networks: Challenges and opportunities, *IEEE Commun. Mag.*, **53** (2015), 75–81. https://doi.org/10.1109/mcom.2015.7180511
14. H. B. McMahan, E. Moore, D. Ramage, S. Hampson, B. Arcas, Communication-efficient learning of deep networks from decentralized data, *arXiv preprint*, (2023), arXiv:1602.05629. https://doi.org/10.48550/arXiv.1602.05629
15. N. Shan, X. Cui, Z. Gao, "DRL+FL": An intelligent resource allocation model based on deep reinforcement learning for mobile edge computing, *Comput. Commun.*, **160** (2020), 14–24. https://doi.org/10.1016/j.comcom.2020.05.037
16. X. Wang, Y. Han, C. Wang, Q. Zhao, X. Chen, M. Chen, In-edge AI: Intelligentizing mobile edge computing, caching and communication by federated learning, *IEEE Network*, **33** (2019), 156–165. https://doi.org/10.1109/mnet.2019.1800286
17. Z. Xu, J. Li, M. Zhang, A surveillance video real-time analysis system based on edge-cloud and FL-YOLO cooperation in coal mine, *IEEE Access*, **9** (2021), 68482–68497. https://doi.org/10.1109/access.2021.3077499
18. S. Ye, L. Zeng, Q. Wu, K. Luo, Q. Fang, X. Chen, Eco-FL: Adaptive federated learning with efficient edge collaborative pipeline training, in *Proceedings of the 51st International Conference on Parallel Processing*, (2022), 1–11. https://doi.org/10.1145/3545008.3545015
19. S. S. Musa, M. Zennaro, M. Libsie, E. Pietrosemoli, Convergence of information-centric networks and edge intelligence for IoV: Challenges and future directions, *Future Internet*, **14** (2022), 192. https://doi.org/10.3390/fi14070192
20. Q. Qi, X. Chen, Robust design of federated learning for edge-intelligent networks, *IEEE Trans. Commun.*, **70** (2022), 4469–4481. https://doi.org/10.1109/tcomm.2022.3175921
21. S. Peng, Y. Yang, M. Mao, D. Park, Centralized machine learning versus federated averaging: A comparison using mnist dataset, *KSII Trans. Internet Inf. Syst.*, **16** (2022), 742–756. https://doi.org/10.3837/tiis.2022.02.020
22. W. Y. B. Lim, N. C. Luong, D. T. Hoang, Y. Jiao, Y. Liang, Q. Yang, et al., Federated learning in mobile edge networks: A comprehensive survey, *IEEE Commun. Surv. Tutorials*, **22** (2020), 2031–2063. https://doi.org/10.1109/COMST.2020.2986024

23. D. C. Nguyen, M. Ding, P. N. Pathirana, A. Seneviratne, J. Li, H. V. Poor, Federated learning for Internet of Things: A comprehensive survey, *IEEE Commun. Surv. Tutorials*, **23** (2021), 1622–1658. https://doi.org/10.1109/COMST.2021.3075439
24. R. Gupta, T. Alam, Survey on federated-learning approaches in distributed environment, *Wireless Pers. Commun.*, **125** (2022), 1631–1652. https://doi.org/10.1007/s11277-022-09624-y
25. L. Witt, M. Heyer, K. Toyoda, W. Samek, D. Li, Decentral and incentivized federated learning frameworks: A systematic literature review, *IEEE Internet Things J.*, **10** (2023), 3642–3663. https://doi.org/10.1109/JIOT.2022.3231363
26. H. Chen, H. Wang, Q. Long, D. Jin, Y. Li, Advancements in federated learning: Models, methods, and privacy, *arXiv preprint*, (2023), arXiv:2302.11466. https://doi.org/10.48550/arXiv.2302.11466
27. M. Al-Quraan, L. Mohjazi, L. Bariah, A. Centeno, A. Zoha, K. Arshad, et al., Edge-native intelligence for 6G communications driven by federated learning: A survey of trends and challenges, *IEEE Trans. Emerging Top. Comput. Intell.*, **7** (2023), 957–979. https://doi.org/10.1109/TETCI.2023.3251404
28. B. Soltani, V. Haghighi, A. Mahmood, Q. Z. Sheng, L. Yao, A survey on participant selection for federated learning in mobile networks, in *ACM Workshop on Mobility in the Evolving Internet Architecture*, (2022), 19–24. https://doi.org/10.1145/3556548.3559633
29. L. Fu, H. Zhang, G. Gao, M. Zhang, X. Liu, Client selection in federated learning: Principles, challenges, and opportunities, *IEEE Internet of Things J.*, **10** (2023), 21811–21819. https://doi.org/10.1109/jiot.2023.3299573.
30. Y. Jin, L. Jiao, Z. Qian, S. Zhang, S. Lu, X. Wang, Resource-efficient and convergence-preserving online participant selection in federated learning, in *IEEE 40th International Conference on Distributed Computing Systems (ICDCS)*, (2020), 606–616. https://doi.org/10.1109/ICDCS47774.2020.00049
31. Y. J. Cho, J. Wang, G. Joshi, Client selection in federated learning: Convergence analysis and power-of-choice selection strategies, *arXiv preprint*, (2020), arXiv:2010.01243. https://doi.org/10.48550/arXiv.2010.01243
32. C. Li, X. Zeng, M. Zhang, Z. Cao, PyramidFL: A fine-grained client selection framework for efficient federated learning, in *Annual International Conference on Mobile Computing and Networking*, (2022), 158–171. https://doi.org/10.1145/3495243.3517017
33. T. Huang, W. Lin, L. Shen, K. Li, A. Y. Zomaya, Stochastic client selection for federated learning with volatile clients, *IEEE Internet of Things J.*, **9** (2022), 20055–20070. https://doi.org/10.1109/jiot.2022.3172113
34. J. Zhao, P. Vandenhove, P. Xu, H. Tao, L. Wang, C. H. Liu, et al., Parallel and memory-efficient distributed edge learning in B5G IoT networks, *IEEE J. Sel. Top. Signal Process.*, **17** (2022), 222–233. https://doi.org/10.1109/jstsp.2022.3223759
35. C. Briggs, Z. Fan, P. Andras, Federated learning with hierarchical clustering of local updates to improve training on non-IID data, in *2020 International Joint Conference on Neural Networks (IJCNN)*, (2020), 1–9. https://doi.org/10.1109/IJCNN48605.2020.9207469
36. W. Q. Shi, S. Zhou, Z. Niu, Device scheduling with fast convergence for wireless federated learning, in *IEEE International Conference on Communications (ICC)*, (2020), 1–6. https://doi.org/10.1109/icc40277.2020.9149138

37. Z. Ji, L. Chen, N. Zhao, Y. Chen, G. Wei, F. R. Yu, Computation offloading for edge-assisted federated learning, *IEEE Trans. Veh. Technol.*, **70** (2021), 9330–9344. https://doi.org/10.1109/tvt.2021.3098022
38. S. Wu, H. Xue, L. Zhang, Q-learning-aided offloading strategy in edge-assisted federated learning over industrial IoT, *Electronics*, **12** (2023), 1706. https://doi.org/10.3390/electronics12071706
39. C. Yu, S. Shen, K. Zhang, Z. Hai, Y. Shi, Energy-aware device scheduling for joint federated learning in edge-assisted internet of agriculture things, in *IEEE Wireless Communications and Networking Conference (WCNC)*, (2022), 1140–1145. https://doi.org/10.1109/wcnc51071.2022.9771547
40. X. Yao, T. Huang, R. X. Zhang, R. Li, L. Sun, Federated learning with unbiased gradient aggregation and controllable meta updating, *arXiv preprint*, (2020), arXiv:1910.08234. https://doi.org/10.48550/arXiv.1910.08234
41. A. R. Elkordy, A. S. Avestimehr, HeteroSAg: Secure aggregation with heterogeneous quantization in federated learning, *IEEE Trans. Commun.*, **70** (2022), 2372–2386. https://doi.org/10.1109/tcomm.2022.3151126
42. C. H. Hu, Z. Chen, E. G. Larsson, Device scheduling and update aggregation policies for asynchronous federated learning, *arXiv preprint*, (2021), arXiv:2107.11415. https://doi.org/10.48550/arXiv.2107.11415
43. L. Wang, W. Wang, B. Li, CMFL: Mitigating communication overhead for federated learning, in *IEEE 39th International Conference on Distributed Computing Systems (ICDCS)*, (2019), 954–964. https://doi.org/10.1109/ICDCS.2019.00099
44. S. Truex, N. Baracaldo, A. Anwar, T. Steinke, H. Ludwig, R. Zhang, et al., A hybrid approach to privacy-preserving federated learning, in *ACM Workshop on Artificial Intelligence and Security*, (2019), 1–11. https://doi.org/10.1145/3338501.3357370
45. P. Liu, S. Xie, Z. Shen, H. Wang, Enhancing location privacy through P2P network and caching in anonymizer, *KSII Trans. Internet Inf. Syst.*, **16** (2022), 1653–1670. https://doi.org/10.3837/tiis.2022.05.013
46. Y. Zhu, C. Liu, Y. Zhang, W. You, Research on 5G core network trust model based on NF interaction behavior, *KSII Trans. Internet Inf. Syst.*, **16** (2022), 3333–3354. http://doi.org/10.3837/tiis.2022.10.007
47. Network simulation version3, 2008. Available from: https://www.nsnam.org/.
48. G. F. Riley, T. R. Henderson, The ns-3 network simulator, in *Modeling and Tools for Network Simulation*, Springer, (2021), 15–34. https://doi.org/10.1007/978-3-642-12331-3_2
49. Gawłowicz, A. Zubow, ns3-gym: Extending OpenAI gym for networking research, *arXiv preprint*, (2018), arXiv:1810.03943. https://doi.org/10.48550/arXiv.1810.03943
50. Network simulation version2, 1997. Available from: https://www.isi.edu/nsnam/ns/.
51. Mininet: Network emulator/simulator, 2010. Available from: http://mininet.org/.
52. Mininet WiFi, 2021. Available from: https://mininet-wifi.github.io/.
53. MATLAB. Available from: https://www.mathworks.com/help/index.html?s_tid=CRUX_lftnav.
54. OMNET++. Available from: https://omnetpp.org/download/models-and-tools.
55. OpenDaylight. Available from: https://www.opendaylight.org.
56. Floodlight. Available from: https://github.com/floodlight/floodlight.
57. Ryu-Controller. Available from: https://ryusdn.org/index.html.
58. OpenStack. Available from: https://www.openstack.org/.

59. Iperf. Available from: https://iperf.fr/.
60. S. Avallone, S. Guadagno, D. Emma, A. Pescapé, G. Ventre, D-ITG distributed internet traffic generator, in *First International Conference on the Quantitative Evaluation of Systems*, *2004. QEST 2004. Proceedings*, (2004), 316–317. https://doi.org/10.1109/qest.2004.1348045
61. Open network foundation. Available from: https://opennetworking.org/.
62. D. J. Beutel, T. Topal, A. Mathur, X. Qiu, T. Parcollet, P. de Gusmao, et al., Flower: A friendly federated learning research framework, *arXiv preprint*, (2022), arXiv:2007.14390. https://doi.org/10.48550/arXiv.2007.14390
63. C. He, S. Li, J. So, X. Zeng, M. Zhang, H. Wang, et al., FedML: A research library and benchmark for federated machine learning, *arXiv preprint*, (2020), arXiv:2007.13518. https://doi.org/10.48550/arXiv.2007.13518
64. FederatedAi/FATE. Available from: https://github.com/FederatedAI/FATE.
65. Tensorflow/federated. Available from: https://github.com/tensorflow/federated.
66. A. Ziller, A. Trask, A, Loardo, B. Wagner, J. Nounahon, J. Passerat-Palmach, et al., PySyft: A library for easy federated learning, in *Federated Learning Systems*, Springer, (2021), 111–139. https://doi.org/10.1007/978-3-030-70604-3_5
67. M. H. Garcia, A. Manoel, D. M. Diaz, F. Mireshghallah, R. Sim, D. Dimitriadis, Flute: A scalable, extensible framework for high-performance federated learning simulations, *arXiv preprint*, (2022), arXiv:2203.13789. https://doi.org/10.48550/arXiv.2203.13789
68. E. Ekaireb, X. Yu, K. Ergun, Q. Zhao, K. Lee, M. Huzaifa, et al., ns3-fl: Simulating federated learning with ns-3, (2022), 99–104. https://doi.org/10.1145/3532577.3532591
69. H. Ludwig, N. Baracaldo, G. Thomas, Y. Zhou, A. Anwar, S. Rajamoni, et al., IBM federated learning: An enterprise framework white paper V0.1, *arXiv preprint*, (2020), arXiv:2007.10987. https://doi.org/10.48550/arXiv.2007.10987
70. G. Ulm, E. Gustavsson, M. Jirstrand, Functional federated learning in Erlang (ffl-erl), in *Functional and Constraint Logic Programming*, Springer, (2018), 162–178. https://doi.org/10.1007/978-3-030-16202-3_10
71. M. Daole, A. Schiavo, J. Bárcena, P. Ducange, F. Marcelloni, A. Renda, OpenFL-XAI: Federated learning of explainable artificial intelligence models in Python, *SoftwareX*, **23** (2023), 101505. https://doi.org/10.1016/j.softx.2023.101505
72. B. Knott, S. Venkataraman, A. Hannun, S. Sengupta, M. Ibrahim, L. van der Maaten, CrypTen: Secure multi-party computation meets machine learning, *arXiv preprint*, (2022), arXiv:2109.00984. https://doi.org/10.48550/arXiv.2109.00984
73. Y. Xie, Z. Wang, D. Chen, D. Gao, L. Yao, W. Kuang, et al., FederatedScope: A flexible federated learning platform for heterogeneity, in *Proceedings of the VLDB Endowment*, (2023), 1059–1072. https://doi.org/10.14778/3579075.3579081
74. H. R. Roth, Y. Chen, Y. Wen, I. Yang, Z. Xu, Y. Hsieh, et al., Nvidia flare: Federated learning from simulation to real-world, *arXiv preprint*, (2023), arXiv:2210.13291. https://doi.org/10.48550/arXiv.2210.13291
75. W. Zhuang, X. Gan, Y. Wen, S. Zhang, EasyFL: A low-code federated learning platform for dummies, *IEEE Internet of Things J.*, **9** (2022), 13740–13754. https://doi.org/10.1109/jiot.2022.3143842

76. S. Caldas, S. Duddu, P. Wu, T. Li, J. Konecny, H. B. McMahan, et al., LEAF: A benchmark for federated settings, *arXiv preprint*, (2019), arXiv:1812.01097. https://doi.org/10.48550/arXiv.1812.01097
77. PaddlePaddle/PaddleFL. Available from: https://github.com/PaddlePaddle/PaddleFL.
78. L. Sani, P. Porto, A. lacob, W. Zhao, X. Qiu, Y. Gao, et al., IBM federated learning: An enterprise framework white paper V0.1, *arXiv preprint*, (2020), arXiv: 2007.10987v1. https://doi.org/10.48550/arXiv.2007.10987
79. P. Tam, S. Math, C. Nam, S. Kim, Adaptive resource optimized edge federated learning in real-time image sensing classifications, *IEEE J. Sel. Top. Appl. Earth Obs. Remote Sens.*, **14** (2021), 10929–10940. https://doi.org/ 10.1109/JSTARS.2021.3120724
80. V. Balasubramanian, M. Aloqaily, M. Reisslein, A. Scaglione, Intelligent resource management at the edge for ubiquitous IoT: An SDN-based federated learning approach, *IEEE Network*, **35** (2021), 114–121. https://doi.org/10.1109/MNET.011.2100121
81. R. Uddin, S. Kumar, SDN-based federated learning approach for satellite-iot framework to enhance data security and privacy in space communication, in *2022 IEEE International Conference on Wireless for Space and Extreme Environments (WiSEE)*, (2022), 71–76. https://doi.org/10.1109/WiSEE49342.2022.9926943
82. V. Balasubramanian, M. Aloqaily, M. Reisslein, FedCo: A federated learning controller for content management in multi-party edge systems, in *2021 International Conference on Computer Communications and Networks (ICCCN)*, (2021), 1–9. https://doi.org/10.1109/ICCCN52240.2021.9522153
83. A. R. Mahmod, G. Caliciuri, P. Pace, A. Iera, Improving the quality of federated learning processes via software defined networking, in *International Workshop on Networked AI Systems (NetAISys'23)*, (2023), 1–6. https://doi.org/10.1145/3597062.3597281
84. G. Li, J. Wu, S. Li, W. Yang, C. Li, Multi-tentacle federated learning over software-defined industrial internet of things against adaptive poisoning attacks, *IEEE Trans. Ind. Inf.*, **19** (2022), 1260–1269. https://doi.org/10.1109/tii.2022.3173996
85. L. Chen, H. Tang, Y. Zhao, W. You, K. Wang, A privacy-preserving and energy-efficient offloading algorithm based on lyapunov optimization, *KSII Trans. Internet Inf. Syst.*, **16** (2022), 2490–2506. https://doi.org/10.3837/tiis.2022.08.002
86. K. M. M. Fathima, M. Suganthi, N. Santhiyakumari, Enhancing the quality of service by GBSO splay tree routing framework in wireless sensor network, *KSII Trans. Internet Inf. Syst.*, **17** (2023), 2188–2208. https://doi.org/10.3837/tiis.2023.08.013
87. P. Tam, S. Math, S. Kim, Intelligent massive traffic handling scheme in 5G bottleneck backhaul networks, *KSII Trans. Internet Inf. Syst.*, **15** (2021), 874–890. https://doi.org/10.3837/tiis.2021.03.004
88. X. Huang, Z. Chen, Q. Chen, J. Zhang, Federated learning based QoS-aware caching decisions in fog-enabled internet of things networks, *Digital Commun. Networks*, **9** (2023), 580–589. https://doi.org/10.1016/j.dcan.2022.04.022
89. P. Tam, S. Math, S. Kim, Optimized multi-service tasks offloading for federated learning in edge virtualization, *IEEE Trans. Network Sci. Eng.*, **9** (2022), 4363–4378. https://doi.org/10.1109/TNSE.2022.3200057

90. J. Xu, J. Lin, Y. Li, Z. Xu, MultiFed: A fast converging federated learning framework for services QoS prediction via cloud–edge collaboration mechanism, *Knowledge-Based Syst.*, **268** (2023), 110463. https://doi.org/10.1016/j.knosys.2023.110463
91. V. Gugueoth, S. Safavat, S. Shetty, Security of internet of things (IoT) using federated learning and deep learning-recent advancements, issues and prospects, *ICT Express*, **9** (2023), 941–960. https://doi.org/10.1016/j.icte.2023.03.006
92. S. Zarandi, H. Tabassum, Federated double deep Q-learning for joint delay and energy minimization in IoT networks, in *IEEE International Conference on Communications Workshops (ICC Workshops)*, (2021), 1–6. https://doi.org/10.1109/iccworkshops50388.2021.9473821
93. Y. Ren, A. Guo, C. Song, Multi-slice joint task offloading and resource allocation scheme for massive mimo enabled network, *KSII Trans. Internet Inf. Syst.*, **17** (2023), 794–815. https://doi.org/10.3837/tiis.2023.03.007
94. Y. Xu, H. Zhou, J. Chen, T. Ma, S. Shen, Cybertwin assisted wireless asynchronous federated learning mechanism for edge computing, in *IEEE Global Communications Conference (GLOBECOM)*, (2021), 1–6. https://doi.org/10.1109/globecom46510.2021.9685076
95. A. Alferaidi, K. Yadav, Y. Alharbi, W. Viriyasitavat, S. Kautish, G. Dhiman, Federated learning algorithms to optimize the client and cost selections, *Math. Probl. Eng.*, **2022** (2022), 1–9. https://doi.org/10.1155/2022/8514562
96. S. Tang, W. Zhou, L. Chen, L. Lai, J. Xia, L. Fan, Battery-constrained federated edge learning in UAV-enabled IoT for B5G/6G networks, *Phys. Commun.*, **47** (2021), 101381. https://doi.org/10.1016/j.phycom.2021.101381
97. D. J. Han, M. Choi, J. Park, J. Moon, FedMes: Speeding up federated learning with multiple edge servers, IEEE *J. Sel. Areas Commun.*, **39** (2021), 3870–3885. https://doi.org/10.1109/JSAC.2021.3118422
98. W. Liu, L. Chen, Y. Chen, W. Zhang, Accelerating federated learning via momentum gradient descent, *IEEE J. Sel. Areas Commun.*, **31** (2020), 1754–1766. https://doi.org/10.1109/TPDS.2020.2975189
99. R. Chen, D. Shi, X. Qin, D. Liu, M. Pan, S. Cui, Service delay minimization for federated learning over mobile devices, *IEEE J. Sel. Areas Commun.*, **41** (2023), 990–1006. https://doi.org/10.1109/JSAC.2023.3242711
100. S. Caldas, J. Konečny, H. B. McMahan, A. Talwalkar, Expanding the reach of federated learning by reducing client resource requirements, *arXiv preprint*, (2019), arXiv:1812.07210. https://doi.org/10.48550/arXiv.1812.07210
101. M. Chen, Z. Yang, W. Saad, C. Yin, H. V. Poor, S. Cui, A joint learning and communications framework for federated learning over wireless networks, *IEEE Trans. Wireless Commun.*, **20** (2021), 269–283. https://doi.org/10.1109/TWC.2020.3024629
102. C. Ma, Distributed optimization with arbitrary local solvers, *Optim. Methods Software*, **32** (2017), 813–848. https://doi.org/10.1080/10556788.2016.1278445
103. X. Yao, C. Huang, L. Sun, Two-stream federated learning: Reduce the communication costs, in *2018 IEEE Visual Communications and Image Processing (VCIP)*, (2018), 1–4. https://doi.org/10.1109/VCIP.2018.8698609
104. B. Luo, X. Li, S. Wang, J. Huang, L. Tassiulas, Cost-effective federated learning in mobile edge networks, *IEEE J. Sel. Areas Commun.*, **39** (2021), 3606–3621. https://doi.org/10.1109/JSAC.2021.3118436

105. J. Konečný, H. B. McMahan, F. X. Yu, P. Richtárik, A. T. Suresh, D. Bacon, Federated learning: Strategies for improving communication efficiency, *arXiv preprint*, (2019), arXiv:1610.05492. https://doi.org/10.48550/arXiv.1610.05492
106. J. Xu, H. Wang, L. Chen, Bandwidth allocation for multiple federated learning services in wireless edge networks, *IEEE Trans. Wireless Commun.*, **21** (2022), 2534–2546. https://doi.org/10.1109/TWC.2021.3113346
107. A. K. Abasi, M. Aloqaily, M. Guizani, Grey wolf optimizer for reducing communication cost of federated learning, in *GLOBECOM 2022 - 2022 IEEE Global Communications Conference*, (2022), 1049–1054. https://doi.org/10.1109/GLOBECOM48099.2022.10001681
108. D. Gurung, S. R. Pokhrel, G. Li, Quantum federated learning: Analysis, design and implementation challenges, *arXiv preprint*, (2023), arXiv:2306.15708. https://doi.org/10.48550/arXiv.2306.15708
109. N. Bouacida, P. Mohapatra, Vulnerabilities in federated learning, *IEEE Access*, **9** (2021), 63229–63249. https://doi.org/10.1109/ACCESS.2021.3075203
110. F. K. Dankar, K. E. Emam, Practicing differential privacy in health care: A review, *IEEE Intell. Inf. Bull.*, **6** (2013), 35–67. https://dl.acm.org/doi/10.5555/2612156.2612159
111. V. Mothukuri, R. M. Parizi, S. Pouriyeh, Y. Huang, A. Dehghantanha, G. Srivastava, A survey on security and privacy of federated learning, *Future Gener. Comput. Syst.*, **115** (2020), 619–640. https://doi.org/10.1016/j.future.2020.10.007