Federated Learning for Internet of Things: Recent Advances, Taxonomy, and Open Challenges

Latif U. Khan, Walid Saad, *Fellow, IEEE*, Zhu Han, *Fellow, IEEE*, Ekram Hossain, *Fellow, IEEE*, and Choong Seon Hong, *Senior Member, IEEE*

***Abstract*—The Internet of Things (IoT) will be ripe for the deployment of novel machine learning algorithm for both network and application management. However, given the presence of massively distributed and private datasets, it is challenging to use classical centralized learning algorithms in the IoT. To overcome this challenge, federated learning can be a promising solution that enables on-device machine learning without the need to migrate the private end-user data to a central cloud. In federated learning, only learning model updates are transferred between end-devices and the aggregation server. Although federated learning can offer better privacy preservation than centralized machine learning, it has still privacy concerns. In this paper, first, we present the recent advances of federated learning towards enabling federated learning-powered IoT applications. A set of metrics such as sparsification, robustness, quantization, scalability, security, and privacy, is delineated in order to rigorously evaluate the recent advances. Second, we devise a taxonomy for federated learning over IoT networks. Finally, we present several open research challenges with their possible solutions.**

***Index Terms*—Federated learning, Internet of Things, wireless networks.**

I. Introduction

INTERNET of Things (IoT) applications can lead to the true realization of smart cities [1]. Smart cities can offer several critical applications such as intelligent transportation, smart industries, healthcare, and smart surveillance, among others [2]–[5]. To successfully deploy these smart services, a massive number of IoT devices are required [6]–[8]. According to statistics, the number of IoT devices will reach 125 billion by 2030 [9]. These IoT devices will generate enormous amount of data. A recent report in [10] revealed that the data generated by IoT devices will reach 79.4 zettabytes (ZB) by 2025. Such a remarkable increase in both IoT network size and accompanying data volume open up attractive opportunities for artificial intelligence [11]. To this end, one can use centralized machine learning schemes to enable these smart IoT applications. However, centralized machine learning techniques suffer from the inherent issue of users' privacy leakage because of the need to transfer the end-devices data to a centralized third party server for training. Furthermore, centralized machine learning might not be feasible when the data size is very large (e.g., astronomical data [12]) and located at multiple locations [13], [14].

To cope with the aforementioned challenge of large, distributed datasets, we can use distributed machine learning to distribute the machine learning workload to multiple machines [15]–[18]. These multiple machines can then train several models at distributed locations in parallel to create a machine learning model. Distributed learning enables high-performance computing via parallel computation of a machine learning model. Two types of approaches such as data-parallel and model-parallel approaches can be used for distributed machine learning. In a data-parallel approach, the whole data is divided among a set of distributed servers, while each server using the same model for training. The model-parallel approach uses exactly the same data for all servers to train distinct portions of a machine learning model [14]. The model-parallel approach might not be suitable for some applications because all the machine learning model can not be divided into parts.

Although distributed machine learning offers parallel computation of models at geographically distributed locations, it does not explicitly addressed some more practical challenges of data and system heterogeneity [19]. Here, system heterogeneity refers to variations in computational resources (CPU-cycles/sec), communication resources, and storage capacity of end-devices, whereas data heterogeneity deals with non-independent and identical distribution (non-IID) of data [20], [21]. Additionally, traditional distributed machine learning algorithms that rely on the model-parallel or the data-parallel approach do not truly preserve the privacy of users. These distributed machine learning schemes use training of machine learning models at distributed servers that rely on data from multiple end-users, and thus suffers from privacy leakage issue. To ensure privacy preservation

Manuscript received September 22, 2020; revised March 12, 2021 and April 23, 2021; accepted May 31, 2021. Date of publication June 18, 2021; date of current version August 23, 2021. This work was supported by the National Research Foundation of Korea (NRF) grant funded by the Korea Government (MSIT) under Grant 2020R1A4A1018607. *(Corresponding author: Choong Seon Hong.)*

Latif U. Khan and Choong Seon Hong are with the Department of Computer Science and Engineering, Kyung Hee University, Yongin 17104, South Korea (e-mail: cshong@khu.ac.kr).

Walid Saad is with the Wireless@VT, Bradley Department of Electrical and Computer Engineering, Virginia Tech, Blacksburg, VA 24061 USA, and also with the Department of Computer Science and Engineering, Kyung Hee University, Seoul 02447, South Korea.

Zhu Han is with the Electrical and Computer Engineering Department and the Computer Science Department, University of Houston, Houston, TX 77004 USA, and also with the Department of Computer Science and Engineering, Kyung Hee University, Seoul 02447, South Korea.

Ekram Hossain is with the Department of Electrical and Computer Engineering, University of Manitoba, Winnipeg, MB R3T 5V6, Canada.

Digital Object Identifier 10.1109/COMST.2021.3090430

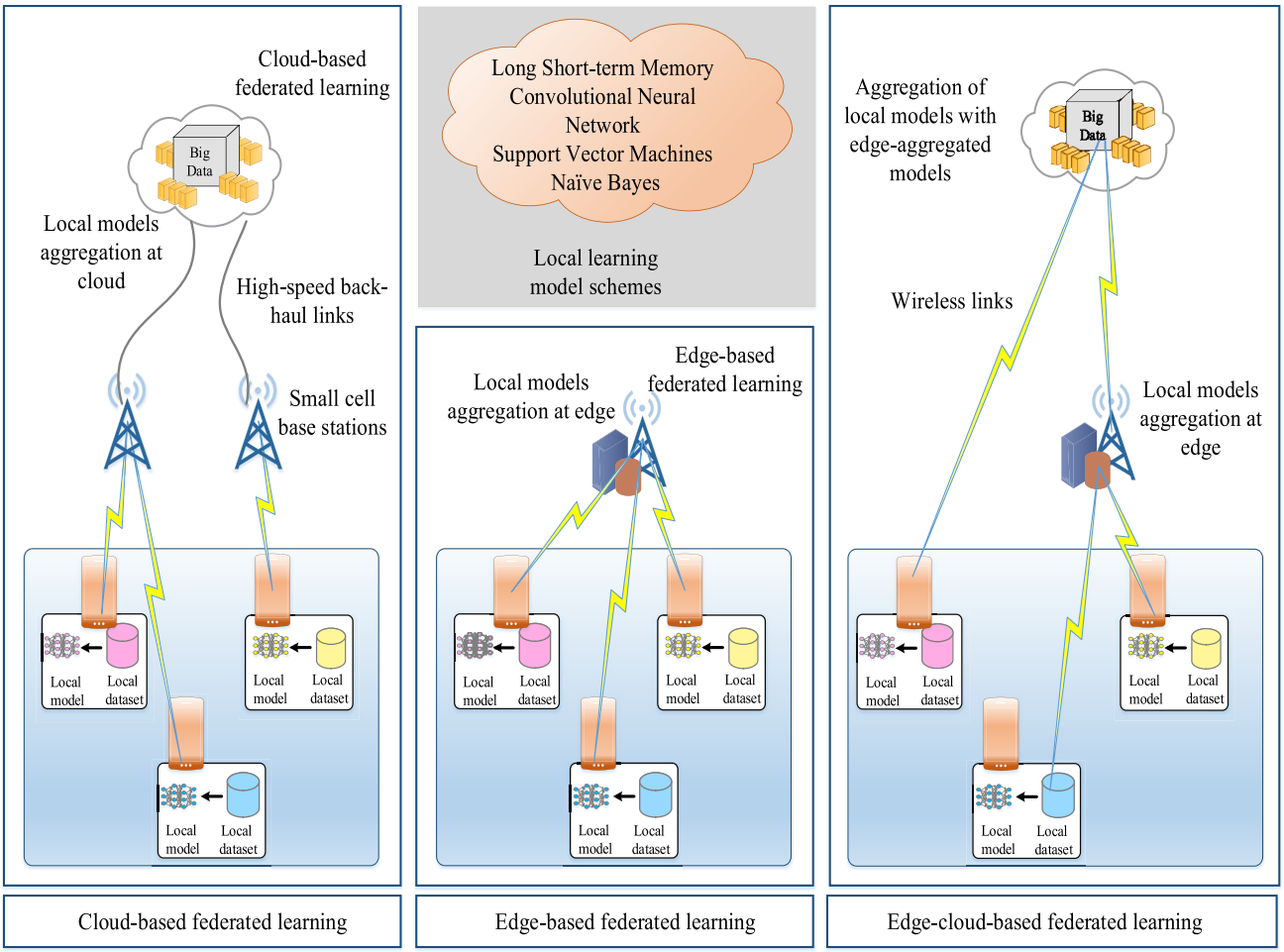


Fig. 1. An overview of federated learning for IoT networks.

while considering system and data heterogeneity in machine learning, federated learning[1] has been introduced [19]. In contrast to traditional machine learning schemes, federated learning does not require migration of data from end-devices to a centralized edge/cloud server for training. The process of federated learning involves local learning model computation at end-devices, which is followed by sending the local learning model parameters to the edge/ cloud server for global model aggregation. Finally, the global model parameters are sent back to the end-devices [22]. To implement federated learning for IoT networks, we will face some challenges. Computing local learning model for federated learning requires significant computational power and backup energy. Additionally, implementing federated learning for IoT applications where the IoT sensors generate fewer data seems challenging [23]–[25]. To train a local learning model, a sufficient amount of data is required for better learning accuracy [26]–[28]. Therefore, training a local learning model with better accuracy for computing resource-constrained devices with less data is challenging. To address this challenge one can migrate the data from resource-constrained IoT sensors (e.g., temperature monitoring sensors) to the nearby trustworthy edge for local model training [29]. On the other hand, autonomous driving cars generate a significant amount of data (i.e., $4,000$ gigaoctet) everyday [30]. Therefore, sending everyday autonomous driving cars will consume more communication resources. One can only send a local learning model to the aggregation server that will consume fewer communication resources than the whole data. Additionally, autonomous cars have more computational power with sufficient energy. Therefore, federated learning can be easily implemented for autonomous driving cars from the perspective of resource usage.

An overview of federated learning for IoT networks is presented in Fig. 1. First of all, end-devices iteratively compute the local learning models using their local datasets. After local model computation, the local learning model updates are sent by end-devices to the aggregation server for the global aggregation. Generally, aggregation can be performed in federated learning either at the edge or cloud. Therefore, we can say that federated learning can be either edge-based or cloud-based depending on the location of the global model aggregation [27]. Edge-based federated learning performs

[1]Federated learning itself has privacy concerns and does not completely guarantee privacy.

global model aggregation at the edge server, whereas cloud-based federated learning performs global federated learning model aggregation at the remote cloud. The edge server serves a small geographical area with a limited number of devices. Therefore, edge-based federated learning is preferable for use in applications that have local context-awareness and require specialized machine learning models for a set of users located in close vicinity to each other [26]. For instance, consider the training of keyboard keyword suggestion model for a language of a particular region using federated learning. There can two ways to train such type of model: a specialized model for the regional language via edge-based federated learning and a generalized model for languages of different regions via cloud-based federated learning. A model using edge-based federated can offer more promising results in such a scenario than cloud-based federated learning [27]. On the other hand, the cloud serves a large number of users located over a large distributed area. Therefore, federated learning that uses aggregation at the cloud (i.e., cloud-based federated learning) is more suitable for training more generalized models than edge-based federated learning by considering a large number of geographically distributed users. It must be noted that we derive the use of terms *edge-based federated learning* and *cloud-based federated learning* from [27]. Other than edge-based federated learning and cloud-based federated learning, we can jointly use both edge and cloud servers for the aggregation of learning models. We refer to federated learning that uses aggregation at both cloud and edge as edge-cloud-based federated learning [31]. Communication with an edge is less expansive in terms of communication resources consumption and latency than the remote cloud [2]. Therefore, end-devices can easily communicate with the edge server that will perform aggregation of local learning models to yield an edge-aggregated learning model. On the other hand, edge server has generally small coverage area and has a limited capacity to serve end-devices. Therefore, few of the end-devices located in a different geographical area as that of the edge server will directly communicate with the cloud. The edge-cloud-based federated learning will perform aggregation of local learning models of these devices with the edge-aggregated learning models. The main advantage of edge-cloud-based federated learning lies in enabling the participation of more devices in the learning process, and thus improves federated learning performance [19], [27], [31].

Although federated learning enables on-device machine learning, it suffers from security, robustness, and resource (both computational and communication) optimization challenges. To truly benefit from the use of federated learning in IoT networks, there is a need to resolve these challenges. To cope with resource constraints issues, one can develop federated learning protocols based on device-to-device (D2D) communication. An IoT device might not be able to communicate with a central base station (BS) due to communication resource constraints. To enable participation of such IoT devices in federated learning, one can use collaborative federated learning [32]. Collaborative federated learning allows resource-constrained IoT devices to send their local learning models to nearby devices rather than the BS, where local aggregation takes place between the receiving device local model and the received local model. Then, the locally aggregated model is sent to the centralized edge/cloud server for global aggregation. Similar to traditional federated learning, collaborative federated learning can be either edge-based or cloud-based depending on the context of the machine learning model. On the other hand, a malicious aggregation server can infer end-devices information from their learning model parameters, and thus might cause privacy leakage [26], [33]. To cope with this issue, one can use differential privacy-aware federated learning [34]. Next, we discuss market statistics and research trends of federated learning and IoT-networks.

A. Market Statistics and Research Trends

According to statistics, the market of artificial intelligence is expected to grow at a Compound Annual Growth Rate (CAGR) of 33.2% from 27.23 billion U.S. Dollar in 2019 to 266.93 billion U.S. Dollar in 2027 [35]. The key driver of the increase in the artificial intelligence market is the proliferation of technological advancement and data generation. Banking, financial and insurance (BFSI) vertical, IT and telecom, retail, healthcare, automotive, advertising and media, among others are various industries that are mainly contributing to the artificial intelligence market. On the other hand, North America has shown significant growth in the artificial intelligence market in 2019 with the U.S. as the highest contributor. The Asia Pacific is also expected to have significant growth in the artificial intelligence market. China will add a major portion to the artificial intelligence market in the Asia Pacific region.

The IoT market is expected to reach 1463.19 billion U.S. Dollar from 250.72 billion U.S. Dollar in 2027 compared to 2019 at a CAGR of 24.9% [36]. The key drivers of IoT markets are telecommunication, transportation, manufacturing, healthcare, government, retail, BFSI, among others. The highest share is hold by BFSI sector. The highest revenue generated by the Asia Pacific region in 2018 is 98.86 billion U.S. Dollar, which is further expected to lead the IoT market in future. Within Asia Pacific region, China has the highest share. Other than market share of machine learning and IoT, the number of research publications versus year are shown in Fig. 2. It is clear from the Fig. 2 that there is significant increase in number of publications every year for both federated learning and IoT. Based on the above-discussed facts, machine learning and IoT are expected to opens up novel opportunities for research and development in the foreseeable future.

B. Existing Surveys and Tutorials

Numerous surveys and tutorials primarily reviewed federated learning [26], [28], [29], [37]–[39], [41], and [42]. Lim *et al.* presented a comprehensive survey of federated learning for mobile edge networks [26]. First, the authors provided an overview of federated learning. Second, the key implementation challenges with existing solutions and several applications of federated learning in mobile edge networks are discussed. Finally, several open research challenges are presented. In [37], Li *et al.* surveyed implementation challenges of federated learning, existing approaches, and future

TABLE I
SUMMARY OF EXISTING SURVEYS AND TUTORIALS WITH THEIR PRIMARY FOCUS

Reference	Internet of Things	Resource optimization for federated learning	Incentive mechanism for federated learning	Learning algorithm design for federated learning	Taxonomy	Remark
Lim *et al.*, [26]	✗	✓	✓	✓	✗	Federated learning at network edge is considered.
Li *et al.*, [37]	✗	✗	✗	✓	✗	N.A
Li *et al.*, [38]	✗	✗	✗	✓	✗	N.A
Wang *et al.*, [29]	✗	✗	✗	✗	✗	Mainly considered federated learning for caching and computational offloading.
Lyu *et al.*, [39]	✗	✗	✗	✗	✗	Surveyed security in federated learning.
Kairouz *et al.*, [28]	✗	✗	✗	✓	✗	N.A
Khan *et al.*, [40]	✗	✗	✗	✗	✗	The work introduced dispersed federated learning, devise its taxonomy, and presented future directions.
Rahman *et al.*, [41]	✗	✓	✗	✓	✗	Surveyed federated learning design aspects (i.e., learning algorithm design and resource optimization) and open challenges.
Aledhari *et al.*,[42]	✗	✗	✗	✓	✗	Focused mainly on federated learning architecture and enablers.
Our Tutorial	✓	✓	✓	✓	✓	N.A

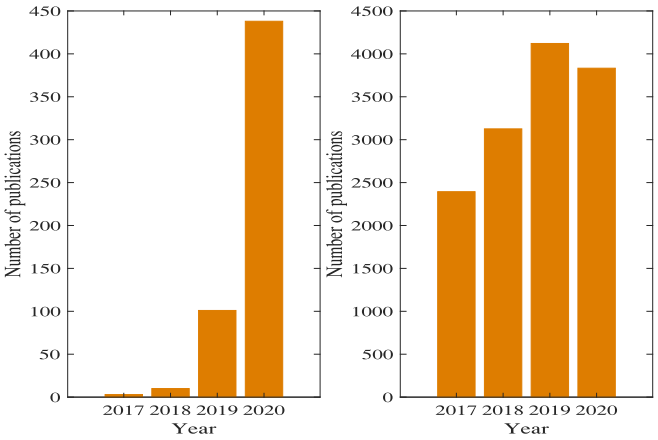


Fig. 2. Year wise publications of (a) federated learning, and (b) Internet of Things [Scopus data, Access month: February 2021].

research directions. The work in [38] presented an overview of federated learning, devised taxonomy, and discussed the existing solutions with their limitations in enabling federated learning. In [29], Wang *et al.* proposed an "In-Edge AI" framework for edge networks to enable efficient collaboration between end-devices and edge servers for learning model updates exchange. They considered two use cases such as edge caching and computation offloading. To efficiently enable these use cases, a double deep Q-learning agent was trained using federated learning. Finally, several open research challenges were presented. In [39], Lyu *et al.* presented a survey of threats to federated learning. In [28], Kairouz *et al.* comprehensively surveyed federated learning, and they discussed basics, privacy preservation, robustness, and ensuring fairness. The authors in [40] presented a vision of dispersed federated learning that is based on true decentralization. Furthermore, a taxonomy and future directions are presented for dispersed federated learning. The authors in [41] primarily surveyed federated learning design aspects (i.e., learning algorithm design and resource optimization) and presented open challenges. Aledhari *et al.* in [42] presented enabling technologies, protocols, and few applications of federated along with open challenges. On the other hand, there are few surveys [43], [44] that have briefly discussed federated learning as a solution to some existing problems (e.g., caching at the network edge). However, their primary perspective was to use machine learning for enabling IoT. In contrast, our tutorial primarily focus on federated learning for IoT.

C. Our Tutorial

Mostly, the existing surveys and tutorials [26], [28], [29], [37]–[42] considered resource optimization, incentive mechanism, and learning algorithm design for federated learning. In contrast (as given in Table I) to work in [26], [28], [29], [37]–[42], we present state-of-the-art advances of federated learning towards enabling IoT applications and devise a taxonomy. Furthermore, the open research

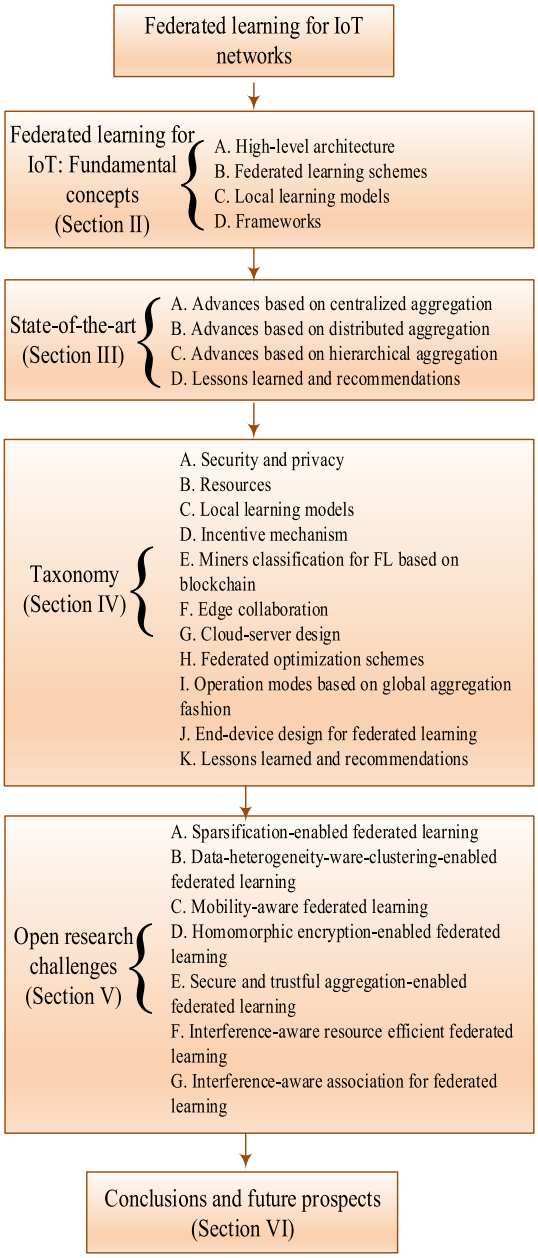


Fig. 3. Overview of our tutorial.

challenges in our paper are significantly different than those in [26], [28], [29], [37]–[40].

The contributions of this tutorial (whose overview is given in Fig. 3) are as follows.

- First, we discuss and critically evaluate the recent federated learning literature, with a focus on the works that are pertinent to IoT applications. To assess these recent works, metrics that serve as key evaluation criteria: Scalability, quantization, robustness, sparsification, security and privacy, are considered. Furthermore, a summary of key contributions and evaluation metrics are presented.
- Second, we derive a taxonomy for federated learning based on different parameters. These parameters include federated optimization schemes, incentive mechanism, security and privacy, operation modes depending on global model aggregation fashion, end-device design, local learning models, resources, miners classification, edge collaboration, and cloud server design.
- Third, we present several key open research challenges with their possible solutions.

The rest of the tutorial is organized as follows: Section II outlines fundamental concepts of federated learning. Recent works of federated learning leading towards IoT applications are presented in Section III. A taxonomy is devised in Section IV using various parameters. Finally, open research challenges with possible solutions are presented in Section V and paper is concluded in Section VI.

II. Federated Learning for IoT: Fundamental Concepts

This section presents a high-level architecture, federated optimization schemes, and various frameworks for federated learning-enabled IoT networks. Next, we briefly discuss various aspects of federated learning-enabled IoT networks.

A. High-Level Architecture

A high-level architecture of federated learning-enabled IoT networks is shown in Fig. 4. The process starts with selection of a set of end-devices according to a suitable criterion [45]. Generally, only a set of devices can participate due to communication resources constraints [46]–[48]. Therefore, we must select devices according to an effective criterion using effective sparsification scheme (step 1). For instance, the criteria can be end-devices with clean datasets, more computational resources, and high-backup power, to enable accurate computation of local learning model within less time [2]. To compute the local learning models using local datasets, we can use various schemes at end-devices, such as feed-forward neural networks (FNN), convolutional neural neural networks (CNN), support vector machines (SVM), and long-short term memory (LSTM), among others (step 2). After computation of the local learning model, we can employ effective quantization scheme to reduce the number of bits used for representation of the local learning model updates (step 3). Various quantization schemes for federated learning were introduced in [49]–[51]. In [49], a lossy federated learning (LFL) scheme based on quantization was proposed. In LFL, both local learning updates and global model updates are quantized by slightly modifying Communication-Efficient stochastic gradient descent (SGD) via Gradient Quantization and Encoding (QSGD) [52] prior to transmission. LFL has shown performance near to federated learning without quantization while reducing communication resources consumption. In another work [50], the authors proposed a secure aggregation scheme using heterogeneous quantization for federated learning. Shlezinger *et al.* [51] presented a subtractive dithered lattice quantization-based federated learning scheme. Moreover, dithered hexgonial lattice quantization has shown less compression error compared to dithered scalar quantization, uniform quantization with random unitary rotation, and subsampling with 3 bits quantizers. Although various lossy quantization schemes of [49]–[51] can offer a tradeoff between accuracy and communication cost for

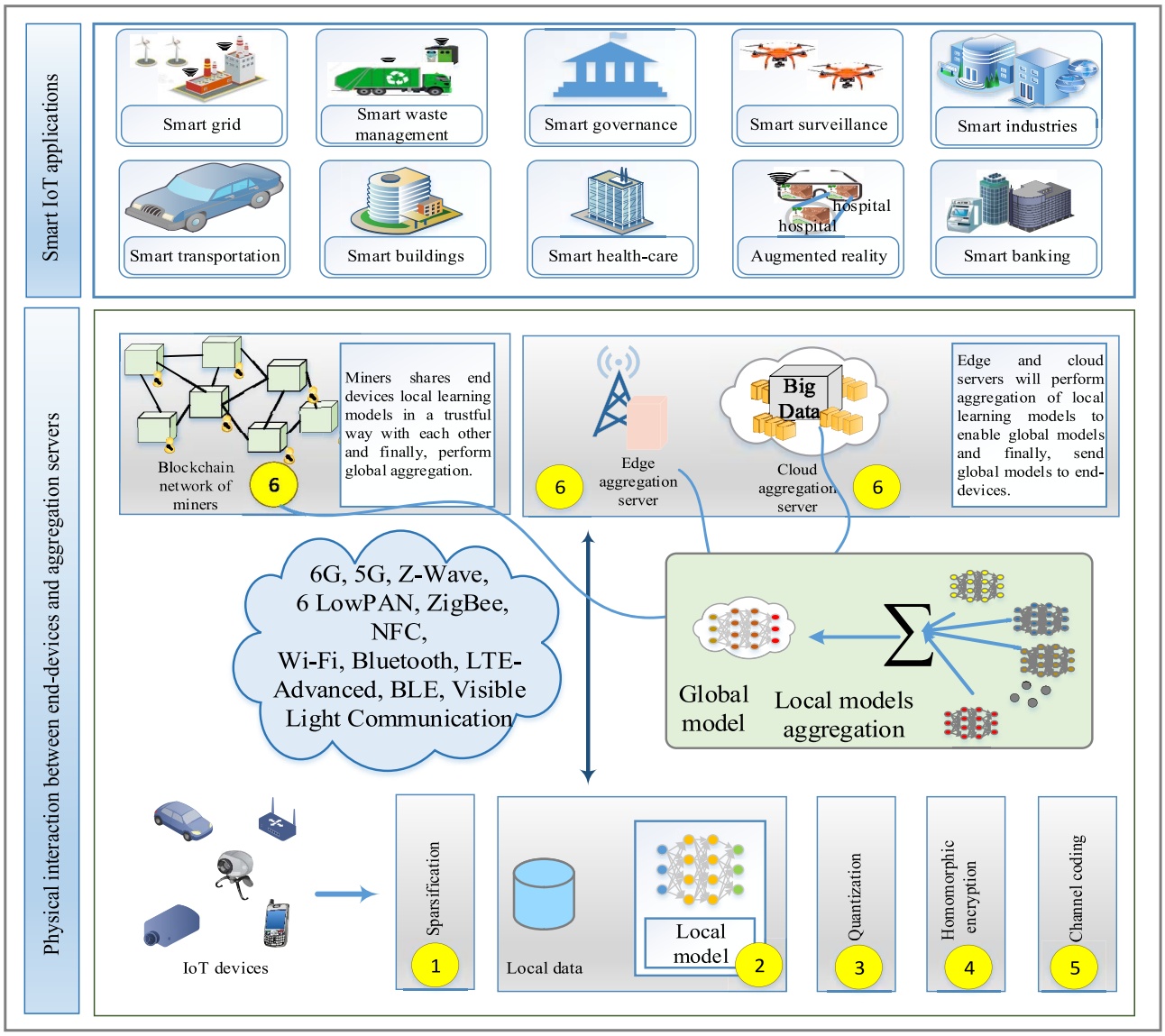


Fig. 4. Architectural overview of federated learning for IoT networks.

federated learning, loss in accuracy due to a quantization must be further reduced.

Next to quantization, we can use some effective encryption algorithm (step 4). A malicious end-device and aggregation server can infer end-devices sensitive information using their local learning model updates [26], [40]. To avoid this kind of privacy leakage, we can use additive homomorphic encryption in federated learning [53], [54]. Although homomorphic encryption will allow aggregation server to perform global model update from encrypted local learning models without the need for decryption, it will add computation and communication cost. The decryption and encryption in case of homomorphic encryption require multiple modular exponentiation and multiplication operations which are complex operations [55]. Moreover, larger ciphertexts are used by homomorphic encryption that will require more communication resources for transmission. To address these issues, the authors in [56] proposed a BatchCrypt scheme that is based on encrypting a batch of quantized gradients. The batch of quantized gradients will be then encrypted and gradient-wise aggregation takes place at the aggregation server. Specifically, BatchCrypt jointly uses batch encoding, quantization, and analytical quantization modeling for improving computation and communication efficiency. One of the main advantage of BatchCrypt is the ability to preserve the aggregated model quality similar to cross-silo federated learning based on homomorphic encryption.

After quantization and encryption, there is a need to use effective channel coding scheme with low overhead and better performance (step 5). The feasible way can be to use Short Block-Length Codes designed for Ultra-Reliable Low Latency Communications (URLLC) [57], [58]. The key design aspects of URLLC codes are latency, reliability, and flexibility. To achieve low latency communication, we should use short packets. Another way to achieve low latency is to use more bandwidth to minimize the latency. Therefore, use of short-block length codes are preferred for URLLC using short packets. Turbo codes, convolutional codes, Bose, Chaudhuri, and

Hocquenghem (BCH), and Low-density parity-check (LDPC) codes can be used for packet error rate improvement. However, the complexity associated with Turbo codes makes them difficult to apply to IoT devices with computation resource constraints [59]. In [57], it is shown that BCH codes exhibits highest reliability under optimal decoding among Turbo codes, LDPC codes, polar codes, and convolutional codes. Therefore, one can use BCH codes for federated learning-based IoT networks where ultra-reliable communication is necessary. After channel coding (step 5), the local learning model updates are transmitted to the aggregation server.

There are many ways to aggregate the local learning models, such as at edge server or cloud server or using miners of a blockchain network [60] and [61]. Aggregation at the edge and cloud server is simple by using averaging in case of FedAvg [27] and then sending back the global learning model updates to end-devices. In case of aggregation via blockchain miners, first of all the miners will receive local model updates from the end-devices. Next, they will share the learning models among themselves and run a blockchain consensus algorithm to update their blocks. After updating the blocks, a global model aggregation takes place at every miner in a distributed manner without the need for centralized aggregation server. Although using blockchain-based miners for global model aggregation in a distributed manner suffers from significant computational complexity due to running a consensus algorithm, it offers the advantage of robustness without the need for a centralized aggregation server. A centralized aggregation server malfunctions due to a security attack or physical damage. Next, we discuss various federated learning schemes depending mainly on various aggregation schemes.

B. Federated Learning Schemes

In this subsection, we present various federated learning schemes. These schemes use different strategies to account for the challenges that are involved in distributed learning systems. The two main challenges in federated learning are:

- System heterogeneity refers to the variations in end-devices features (i.e., computational resource (CPU-cycles/sec), backup power, etc.) and wireless channel uncertainties.
- Data heterogeneity refers to unbalanced local datasets and non-IID data among end-devices.

To show how federated learning model is trained, consider a set $\mathcal{N}$ of N devices. Every device n has a local dataset d_n of k_n data points. The goal of federated learning is to minimize the global loss function f_{FL}, i.e., $\min_{\boldsymbol{\omega}} \frac{1}{K}\sum_{n=1}^{N}\sum_{k=1}^{k_n} f_k(\boldsymbol{\omega})$. Where ω and K represent the global model weights and total data points of all devices involved in learning, respectively. The nature of the loss function f_{FL} depends on problem. For instance, prediction error can be the loss function for prediction of a time series, whereas classification error for classification tasks. One way to train a global machine learning model for a set of devices with local dataset is through stochastic gradient descent (SGD) [62]. However, typically SGD involves one device in training of a local learning model during one communication round that may take long time for convergence. Therefore, one must modify the SGD algorithm for federated setting. In federated SGD, for all end-devices (i.e., fraction of clients $C = 1$) and learning rate η, every client computes $g_n = \bigtriangledown F_n(\boldsymbol{\omega})$ (where $F_n = \frac{1}{K_n}\sum_{k=1}^{k_n} f_k(\boldsymbol{w})$ is computed once using their local dataset d_n). Then all the local models are aggregated and the update $\boldsymbol{\omega}^{t+1} \leftarrow \boldsymbol{\omega}^t - \eta\sum_{n=1}^{N}\frac{k_n}{K} g_n$ is applied. Although federated SGD can enable distributed machine learning without migrating the end-devices local datasets to the centralized server for training, its performance can be further improve if we perform more local iterations before sending the local model updates to an aggregation sever. Overview of FedAvg is given in Algorithm 1 that uses multiple local iterations at devices before sending model updates to the aggregation server [19].

Algorithm 1 FedAvg [19]

Aggregation Server
Weights initialization $\boldsymbol{\omega}^0$
for t=0, 1,..., Global Rounds-1 **do**
  Select $\mathcal{N}_s \leftarrow m = max(C.N)$ clients randomly.
  for For every device $n \in \mathcal{N}_s$, parallel run. **do**
    $\omega_n^{t+1} \leftarrow DeviceUpdate(n, \omega_n^t)$
    $\boldsymbol{\omega}^{t+1} \leftarrow \sum_{n=1}^{N_s}\frac{k_n}{K}\omega_n^{t+1}$
  end for
end for
DeviceUpdate*(n,ω)*
$\mathcal{B} \leftarrow$ Split d_k into batches.
for e=0, 1, .. , Local iterations-1 **do**
  for $b \in \mathcal{B}$ **do**
    $\omega \leftarrow \omega - \eta \bigtriangledown l(\omega, b)$
  end for
end for

Although the FedAvg scheme [19] was the first work to enable distributed learning in environments with main focus on heterogeneous systems parameters, and data distributions, it seems difficult to provably get convergence [28]. This is because FedAvg is based on a simple averaging of local model updates from the end-devices at aggregation server that might not produce better results for heterogeneous environments. The work in [19] has shown that hyper parameters (i.e., local iterations, global iterations, learning rate, etc.) play an important role in convergence of FedAvg. However, for high number of local iterations, the dissimilar local objectives might lead end-devices toward their local optima rather than the global one. Therefore, we can say that FedAvg seems difficult to provably converge for heterogeneous scenarios. Furthermore, setting higher local iterations may cause some of the devices not to complete computing their local learning models within maximum allowed time. Therefore, these devices will not be able to participate in learning process for the given communication round. To cope with these limitations, FedProx was introduced in [63] that is based on adding a proximal term to local model loss function term in FedAvg.

$$\min_{\boldsymbol{\omega}} h_n(\omega; \omega^t) = F_n(\omega) + \frac{\mu}{2}\|\omega - \omega^t\|^2 \tag{1}$$

In (1), the proximal term offers the advantage of better addressing the statistical heterogeneity by restricting the local updates more to a global model, and thus devices heterogeneous data impact will be minimized on the local model. Both FedAvg and FedProx tried to effectively consider the effect of heterogeneous system parameters (i.e., local computational resource) and data heterogeneity (i.e., unbalanced datasets, non-independent and identical distributions), they did not study the effect of wireless channel uncertainties on the performance federated learning. For enabling federated learning over IoT networks, we must carefully handle loss in performance of federated learning due to wireless channel uncertainties. Chen *et al.* in [64] studied the effect of wireless channel uncertainties on the performance of federated learning. They presented a framework for joint learning and communication. Moreover, they formulated a problem to minimize a federated learning cost by jointly optimizing power allocation, resource block allocation, and user selection. In another work, Tran *et al.* analyzed the performance of federated learning over wireless networks [65]. An optimization problem to minimize both energy consumption and computation time was formulated. Moreover, they provided extensive numerical results to analyze the performance of federated learning over wireless networks.

C. Local Learning Models

We present various local learning models used for computing local learning models in federated learning. The type of local learning model used strictly depends on the IoT application considered. CNN, FNN, LSTM, and SVM, among others can be used for local model computation. Table II lists various machine learning algorithms for IoT applications [66]–[77]. Typically, a neural network has an input layer, hidden layers, and the output layer. Increasing the number of hidden layers generally improves the performance but at the cost of computational complexity. On the other hand, the selection of a particular local learning model strictly depends on the application. For instance, CNN is normally considered for IoT tasks that are based on image classification. One can also consider simple fully connected FNN with low complexity for image classification. However, the low complexity local learning model might not perform very well. Coping with this issue, CNN with sufficient layers should be preferably used. It must be noted here that increasing the number of hidden layers does not always guarantee performance improvement [28]. Therefore, we must properly choose network size. Furthermore, activation functions (i.e., determines output of a neural network) must be properly used in various neural networks. Activation functions can be broadly categorized into: binary step functions, linear activation functions, and non-linear activation functions. Within non-linear activation functions, there can be sigmoid, hyperbolic tangent (tanh), rectified linear unit (ReLU), soft-max, among others [78]. A summary of activation functions that can be used in various local learning model is given in Table III. For federated learning, selection of a local learning model with low complexity and better performance is of significant interest. In

TABLE II
LOCAL LEARNING MODELS USED FOR IOT APPLICATIONS

Local learning model	Reference	Primary IoT application
SVM	[71]	Network slicing.
	[66]	Malware detection.
CNN	[67]	Disease diagnosis based on image classification.
	[68]	Caching for infotainment-enabled smart cars.
LSTM	[69]	Traffic prediction in Vehicle-to-Vehicle Communication.
	[70]	Edge caching.
FNN	[72]	Distributed denial of service attacks detectio.n
K-means	[73]	Clustering of sensor networks to minimize packet error rate.
	[74]	Proactive caching for IoT networks.
RNN	[75]	Time series forecasting.
Naive Bayes	[76]	Botnets attack detection for IoT.
	[77]	Anomaly detection for IoT-Fog-Cloud computing model.

contrast to the centralized machine learning, federated learning involves a massive number of end-devices with lower computational capacity than the centralized server used by centralized machine learning. Furthermore, there are energy limitations of the IoT end-devices. Keeping in the mind the aforementioned points, there is a need to use efficient local learning model for federated learning. To effectively choose the neural network size, one can use neural architecture search (NAS) to find the optimal architecture out of possible available architectures [79]. More details on how NAS can be employed for local learning models at the end-devices will be given in Section IV-J.

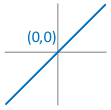
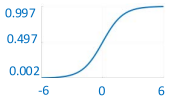
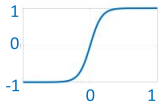
Next to selection of the local learning model, there must be effective technique to optimize the local model weights update to enable faster convergence of the federated learning model. Mostly, SGD was considered for local model weights updating [28]. In SGD, the weights are updated at end-devices using partial derivative of loss function w.r.t weights, which are then multiplied by the learning rate. The learning rate represents the step size of gradient descent. The mini-batch SGD is given by.

$$\omega = \omega - \eta \frac{\partial F}{\partial \omega_t} \tag{2}$$

$$\frac{\partial F}{\partial \omega} \approx \frac{1}{m} \sum_{i \in \mathcal{B}} \frac{\partial f_i}{\partial \omega}, \tag{3}$$

where m denotes the number of elements in a batch. The batch size significantly affects the performance global federated learning model. For FedAvg, using client fraction $C = 0.1$ and batch size 10 generally have shown better results compared to full batch size for MNIST dataset [19]. Other than client fraction C and batch size, learning rate significantly affects the performance of the federated learning global model.

TABLE III
SUMMARY OF COMMONLY USED ACTIVATION FUNCTIONS FOR LOCAL LEARNING MODELS

Activation function	Equation	Derivative	Diagram	Limitation
Unit step function	$f(x) = \begin{cases} 0 & \text{if } x < 0 \\ 0.5 & \text{if } x = 0 \\ 1 & \text{if } x > 0 \end{cases}$	$f'(x) = \delta(t)$		Does not allow multi-value outputs (i.e., classification of inputs into several categories).
Linear function	$f(x) = x$	$f'(x) = 1$		The last layer will be the linear sum of previous layers independent of the number of layers are in a network. Therefore, a neural network with a linear activation function is simply a linear regression model.
Sigmoid function	$f(x) = \frac{1}{1+e^{-x}}$	$f'(x) = f(x)(1 - f(x))$		The change in prediction is not significant for lower and higher values of input. Therefore, it might cause a vanishing gradient problem. Moreover output is not centered at zero and it is computationally expensive.
Hyperbolic tangent	$f(x) = \frac{e^{x}-e^{-x}}{e^{x}+e^{-x}}$	$f'(x) = 1 - f^2(x)$		Similar to sigmoid, the change in prediction is not significant for lower and higher values of input. Additionally, it is computationally expensive.
Rectified linear unit	$f(x) = \begin{cases} x & \text{if } x \geq 0 \\ 0 & \text{if } x < 0 \end{cases}$	$f'(x) = \begin{cases} 1 & \text{if } x \geq 0 \\ 0 & \text{if } x < 0 \end{cases}$		For zero or less than zero inputs, the function becomes zero and does not learn.

Therefore, we must wisely chose the client fraction C, batch size, and learning rate.

D. Frameworks

In this subsection, we discuss various frameworks developed for implementing federated learning over IoT networks [26], [80].

- **PySyft:** PySyft is a python framework that can be used to implement secure and private deep learning [81]. End-devices private data is decoupled during the training process by using Encrypted Computation (like Multi-Party Computation (MPC) and Homomorphic Encryption (HE)), differential privacy, and federated learning. PySyft is primarily based on Pytorch and it retains native Torch interface [26]. For simulation of federated learning using PySyft, end-devices are created as virtual workers. The data is divided into virtual workers. Next, a PointerTensor is used for specification of data storage location and owner. For global federated learning model aggregation, local model updates can be fetched from virtual workers.
- **FedML:** FedML is an open research framework to facilitate development of federated learning [80]. FedML has two main components, such as FedML-core and FedML-API. FedML-core performs separation of model training and distributed communication into two modules, such as distributed communication module and training module. The responsibility of distributed communication module is to carry out low-level communication among various clients, whereas training module is implemented using PyTorch. New algorithms in FedML can be easily implemented using client-oriented programming interface. The primary advantage of FedML is a TopologyManager that can provide a support for many network topologies to implement various federated learning algorithms.
- **TensorFlow Federated:** TensorFlow Federated (TFF) is an open source framework developed by Google for performing experiments on distributed datasets using machine learning and other computations [82]. TFF interfaces can be divided into two types, such as federated core API, and federated learning API. Federated learning API provides high-level interfaces set that enable us to use the existing TensorFlow models. Federated core API is set of lower-level interfaces that can enable new federated learning algorithms using TensorFlow and distributed communication operations.
- **LEAF:** LEAF is a framework for federated settings that can support various applications, such as meta-learning, multi-task learning, and federated learning [83]. LEAF consists of three parts, such as open-source datasets suit,

system and statistical metrics array, and reference implementations set. LEAF offers the modular design that has the advantage of easier implementation of diverse experimental scenarios. The datasets suit has Federated Extended MNIST, Sentiment140, Shakespeare, CelebA, Reddit, and synthetic dataset that is based on modification of synthetic dataset of [84] for making it more challenging for meta-learning methods. Initial implementations of LEAF are Mocha [85], FedAvg [19], and minibatch SGD.

- **Paddle FL:** Paddle FL is an open source framework developed mainly for industry applications [86]. Paddle FL can easily replicate various federated learning algorithms for large scale distributed clusters. Paddle FL will offer federated learning implementation in natural language processing, computer vision, and so on. Furthermore, support for implementation of transfer learning and multi-task learning in federated learning will be provided [87]. Using elastic scheduling of training job on Kubernetes and large scale distributed training of PaddlePaddle's, paddle FL can be easily deployed on full-stack open sourced software.

All of the above-mentioned frameworks present implementation of various federated learning schemes, but they did not effectively considered the effect of wireless channel uncertainties and limited communication resources. IoT applications will suffer from significant performance degradation due to wireless channel uncertainties. Therefore, we should propose novel federated learning frameworks for IoT networks that will enable us to effectively handle wireless channel uncertainties. Furthermore, we can use channel coding schemes to improve the performance of the federated learning over IoT networks. Therefore, the framework for federated learning over IoT networks must provide support for channel coding schemes.

III. State-of-the-Art

This section discusses the recent works in federated learning towards enabling IoT-based smart applications. We critically investigate and rigorously evaluate these recent works, whose summary is given in Tables VI and VII. To rigorously evaluate the recent works, we derive several key metrics from the literature [29], [45], [88]–[100] as follows.

- **Security and privacy:** Although federated learning was developed to preserve the users' privacy, it still faces privacy challenges. For instance, the edge/cloud server can infer the end-devices' sensitive information using their learning model updates, and thus causes the end-device privacy leakage. Furthermore, the malicious user can infer the devices' sensitive information from their local learning models during their exchange with the aggregation server. Beyond inferring the end-devices' sensitive information, the malicious user can alter the learning model updates, and thus prolong the convergence time of a federated learning model. On the other hand, an unauthorized user can access the edge/cloud server and perform false data injection. Therefore, it is necessary to enable federated learning with security and privacy.
- **Scalability:** For a limited communication resource, scalability refers to the ability to incorporate more devices in the federated learning process. Generally, incorporating more devices in federated learning can lead to more accurate results. Scalability in federated learning can be enabled via different schemes such as resource optimization, selection of high-computational power devices, and compression schemes for learning model parameters transmission. An efficient resource optimization scheme for fixed communication resources can enable more devices to participate in the federated learning process, and thus offers better performance. Typically, a set of devices can participate in the federated learning process [26]. Therefore, for synchronous federated learning algorithms that allow fixed local model computation time for all the participating devices, selecting the devices' high-computation power (CPU-cycles/sec) will enable more devices to participate in the federated learning process.
- **Quantization:** The concept of quantization refers to schemes used to reduce the size of the local learning model updates. Minimizing the number of local model updates size results in high throughput, which subsequently minimizes the federated learning convergence time.
- **Robustness:** This deals with the ability of the system to successfully carry out the process of federated learning during edge/cloud server failure. Traditional federated learning based on a centralized edge/cloud server for global aggregation suffers from interruption if the aggregation server stops working due to malfunctioning.
- **Sparsification:** This metric deals with the selection of a set of most suitable devices under communication resource constraints for federated learning. Under communication resource constraints, only a subset of a massive number of devices will be allowed to participate in federated learning. The selection criteria for suitable devices can be less noisy datasets, large size datasets, and data uniqueness. Selecting the most suitable device set results in low convergence time for federated learning. Different from quantization-based schemes that reduce the size of local learning model parameters for all devices before sending to the aggregation server, sparsification-based schemes select only a set of most relevant devices according to a particular criterion.

A summary of the evaluation metrics with advantages is given in Table IV. We use (✓) if the evaluation metric is fulfilled by recent advances and (✗) otherwise. A summary of these recent works is given in Table V. Furthermore, Table VI and Table VII evaluate these works using precise metrics (discussed later). We categorize the recent advances mainly into advances based on centralized aggregation, advances based on distributed aggregation, and advances based on hierarchical aggregation. Fig. 5 shows the overview of various aggregation fashion for federated learning. In centralized aggregation, a single edge or cloud server acts for the aggregation of local learning models of all devices. Distributed aggregation involves multiple aggregation servers that receive local

TABLE IV
EVALUATION METRICS: EXPLANATION, DIMENSION, AND ADVANTAGES

Metric	Explanation	Benefits for federated learning
P_1: **Security and privacy**	This metric deals with malicious user attacks during learning model parameters transmission between end-devices and the aggregation server. Moreover, the malicious aggregation server and end-device can infer other end-devices sensitive information from their local learning model updates. Therefore, we must propose federated learning schemes that offer both security and privacy.	• Users' privacy protection. • Trustful verification of learning model updates. • Secure exchange of learning model updates.
P_2: **Scalability**	Scalability captures to the ability of a federated learning system to incorporate more users during the training process for better accuracy.	• High federated learning accuracy. • Massive connectivity during federated learning process. • Better accuracy due to participation of more users.
P_3: **Quantization**	Quantization refers to the need for minimizing the size of local learning model updates to reduce federated learning convergence time.	• Fast federated learning convergence. • Better accuracy due to participation of more users for fix communication resources.
P_4: **Robustness**	This refers to the ability of a federated learning algorithm to perform the learning process successfully in face of a possible malfunction or failure of the edge/cloud server.	• Cost optimization. • Accurate federated learning model.
P_5: **Sparsification**	Sparsification refers to the selection of most suitable devices among a set of massive numbers of devices according to a specific criteria.	• Lower federated learning convergence time. • High federated learning accuracy.

learning model updates from their associated devices. The received local learning models from various devices are shared among various aggregation servers which are followed by global aggregation. On the other hand, hierarchical aggregation involves aggregations (e.g., at edge servers) before a central global aggregation takes place. Next, we present various advances of federated learning for IoT networks.

A. Advances Based on Centralized Aggregation

Many works such as [29], [45], [64], [88], [89], [93]–[95], [97], and [99] considered federated learning based on a centralized aggregation of end-devices local learning models for various IoT applications. In all these works, end-devices compute their local learning models and send the learning model updates to a centralized aggregation server for computing the global federated learning model. The centralized aggregation server can be either located at the network edge or a remote cloud. In [29], Wang *et al.* proposed an intelligent framework as shown in Fig. 6, namely, In-Edge-AI for edge computing-enabled IoT networks. The authors provided a detailed discussion on how to use Deep Q-Learning agents for two tasks, computation offloading and edge caching. Different use cases of deep Q-learning agent deployment were discussed with their pros and cons. Motivated by the notion of unbalanced, non-IID data, and limited communication resources, federated learning was proposed for the training of deep Q-learning agents. Simulation results showed that federated learning has significantly reduced the training cost for the proposed In-Edge-AI framework. Finally, the authors presented open research challenges regarding the practical deployment of their framework. Although the authors in [29] proposed a detailed framework regarding the implementation of Deep Q-Learning agents based on federated learning, the authors did not properly addressed the issues of security and privacy in their framework. A malicious end-device or an aggregation server can easily infer the end-devices sensitive information from local learning models. Therefore, we must tackle the issues of security and privacy for the In-Edge AI framework. Additionally, Deep Q-Learning agents based on federated learning result in slight loss of performance compared to Deep Q-Learning agents using traditional, centralized machine learning. One possible reason for this performance degradation can be the simple averaging of local learning models. There exist significant variations in the wireless channel and local learning models that must be taken into account while averaging the learning model updates from the edge servers where agents are deployed. One way can be weighted averaging by assigning higher weights to poorly performing nodes so as to minimize the impact of better performing devices on the global model. In another work [88], the authors also considered federated learning at the network edge. They analyzed the convergence bound for federated learning using gradient descent. A novel theoretical convergence bound was derived that considered non-IID nature of the data. Furthermore, based on the derived convergence bound, a control algorithm was proposed that minimizes learning loss for a fixed resource budget. The control algorithm performed learning loss minimization by learning model characteristics,

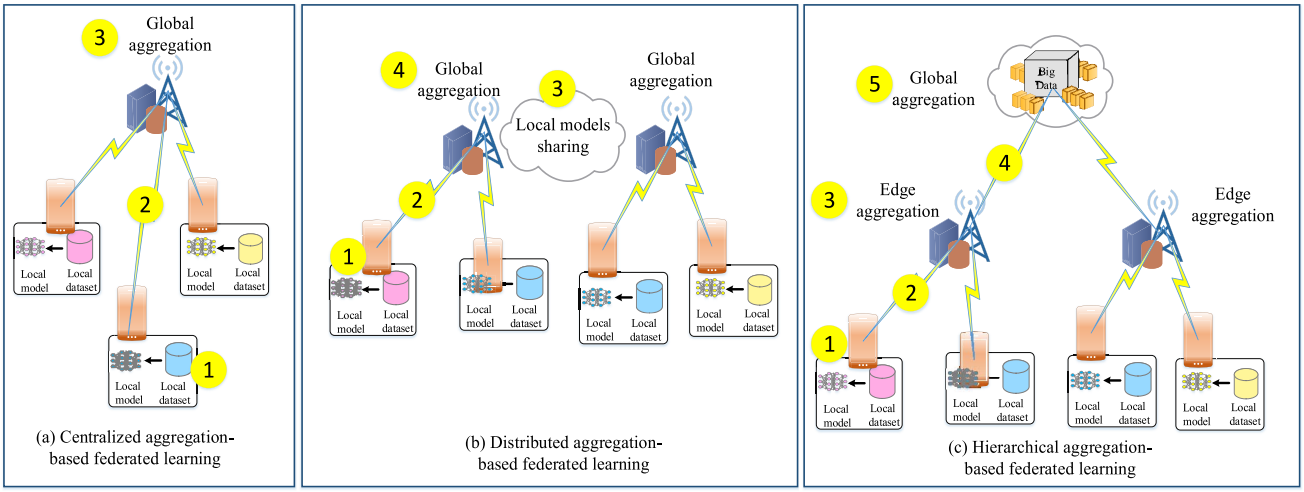


Fig. 5. Federated learning based on: (a) centralized aggregation, (b) distributed aggregation, and (c) hierarchical aggregation.

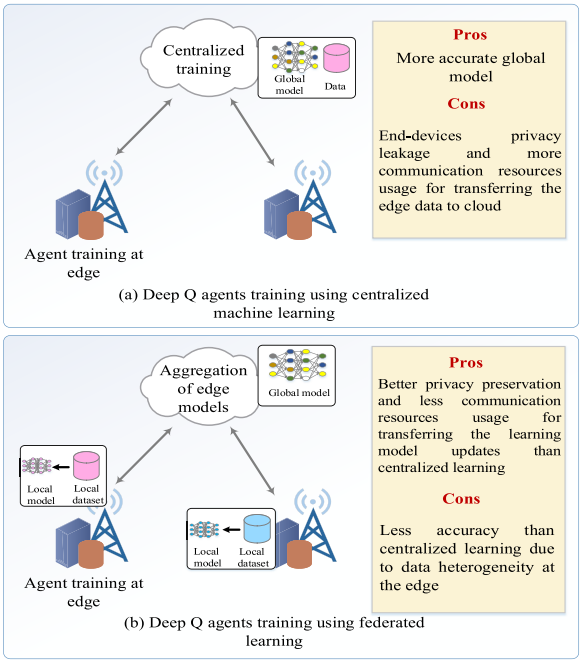


Fig. 6. Deep Q-Learning agents training using (a) centralized training, (b) federated learning [29].

system dynamics, and data distribution. The performance of the proposed control algorithm was evaluated for real datasets using hardware prototype and simulations results. Although the work in [88] presented extensive theoretical convergence analysis for convex loss functions, it does not provide analysis for non-convex loss functions which is more practical for various IoT learning tasks. Additionally, efficient management of resources such as local computation resources and wireless resources must be performed for federated learning over edge networks.

In [64], a framework for joint learning and communication for wireless federated learning was presented. First, the authors extensively studied the performance of federated learning for wireless networks. Then, joint wireless resource allocation, power allocation, and user selection optimization problem was formulated for minimizing the federated learning loss function. For the expected convergence rate of federated learning, a closed-form expression was derived to account for the wireless factors on learning performance. Using the derived closed-form expression, optimal transmit power allocation was performed for a fixed resource block allocation and user selection. Next, resource block allocation and user selection were performed to minimize the federated learning loss function. The framework of [64] can offer scalability by enabling the participation of more users using efficient resource allocation. Moreover, it can offer sparsification by enabling the participation of suitable devices. Although the proposed framework offered several advantages, its performance against the security attacks can be further enhanced by using encryption schemes. Beyond wireless security, the centralized aggregation server might fail due to physical damage, and thus can be suffered from robustness issues.

Beyond the theoretical works on the deployment of federated learning at the network edge in [29], [64], [88], the work in [89] proposed an architecture, namely, CoLearn for federated learning at edge networks. CoLearn was proposed to address the challenges of malicious devices' presence and asynchronous participation of resource-constrained IoT devices [89]. CoLearn uses federated learning framework PySyft and open-source Manufacturer Usage Description (MUD). PySyft is a python framework that can be used to implement federated learning [81]. PySyft uses Pytorch while retaining native Torch interface. Fig. 7 shows the interaction between different components of CoLearn architecture. The CoLearn mainly consists of MUD manager and user policy server, federated learning server, and IoT devices. The IoT devices transmit their MUD-URLs in their DHCP requests to the router. The MUD manager sends valid IoT devices to the federated learning coordinator to minimize the risk of attack. Moreover, the federated learning coordinator

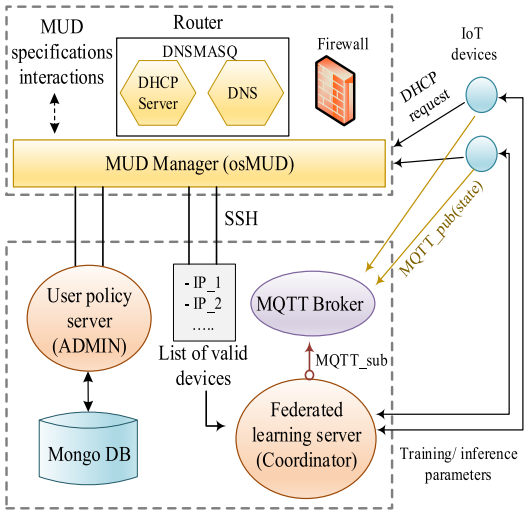


Fig. 7. CoLearn framework [89].

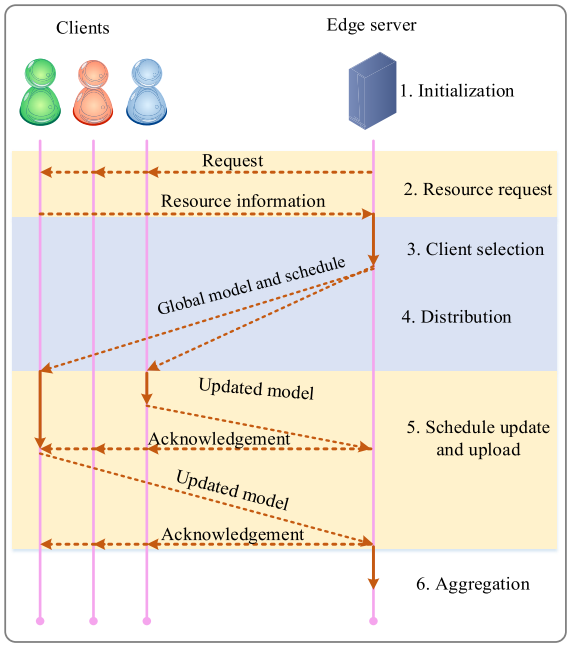


Fig. 8. FedCS protocol steps [45].

performs interaction with devices using WebSockets and MQTT. To enable more flexibility in the design, the user policy server enables a network administrators to enforce rules other than defined by the manufacturer. The proposed architecture showed significant advantages, its performance can be further improved by applying light-weight authentication algorithms at the IoT devices. Another study in [95] demonstrated federated learning over emulated wide-area communications network having features of intermittent, heterogeneous, and dynamic availability of resources. The network is emulated using CORE/EMANE and consists of both mobile and fixed nodes. Any node can be selected as a server that initiates the process of federated learning. The specifications, i.e., as the data type, expected prediction labels, and model architecture are broadcasted to all nodes during the task initialization phase. All the nodes wait for a certain time after the task initialization is done and then start the federated learning process. Any node in a network can leave or enter the learning process. The outdated local learning model parameters of the nodes are ignored and are not considered in the federated learning process. The proposed emulated federated learning environment has advantages of considering intermittent and dynamic behavior of the network resources that make it more feasible for practical applications. In [95], the authors provided an implementation of asynchronous federated learning for a scenario with both mobile and static nodes. However, a random selection of any node to act as an aggregation server might not always result in better performance. There must be an effective criterion for the selection of nodes to acts as a server. One possible criterion can be a selection of a static node as a server. This will enable higher connectivity to other static nodes compared to mobile nodes. However, this approach might not yield promising results when the number of mobile nodes is significantly greater than the static nodes. Therefore, we must propose effective criteria for the selection of nodes as an aggregation server that will result in higher connectivity with other devices involved in federated learning.

The aforementioned works in [29], [64], [88], [89], and [95] provide theoretical analysis pertaining to the deployment of federated learning as well general architecture with implementation for IoT networks. In contrast, the work in [45] particularly focused on a federated learning client selection protocol, namely, FedCS (shown in Fig. 8) that offers an efficient selection of clients for federated learning process based on their available resources. Typically, client selection is performed randomly by a federated learning algorithm that might suffer from an increase in overall federated learning convergence time. For instance, selecting the nodes with the low computational resource (CPU-cycles/sec) results in an increase in overall federated learning time for synchronous federated learning. In FedCS, first of all, the edge computing server requests all the end-devices for their information such as data size, computational capacities, and wireless channel states. Based on the information, a set of nodes is selected for the federated learning process. Then, the edge server distributes the learning model updates to the selected nodes. Finally, all the selected nodes for learning update their local models and send them to the edge server. Simulation results showed that the FedCS mostly attains comparable accuracy within less time compared to traditional federated learning protocols for CIFAR-10 and Fashion-MNIST datasets. The FedCS protocol uses end-devices private information (e.g., the number and classes of its data) by aggregation server, and thus it might suffer from end-devices privacy leakage issue. Additionally, FedCS tried to involve a higher number of end-devices in learning for better performance. However, it does not consider power allocation that can be optimally performed to minimize the transmission latency so that a large number of devices will be able to participate in the synchronous federated learning process. Synchronous federated learning will consider only end-devices local learning models that arrive within a maximum limit of waiting time. On the other hand, a single aggregation edge server is considered in FedCS at

BS. There may be multiple base stations (BSs) and an aggregation server will be at the remote cloud. For this kind of scenarios, one must use computing resource efficient devices and a BS association scheme for minimizing the transmission latency.

To apply federated learning for IoT networks, we must take into account the reliability of end-devices involved in federated learning. A malicious end-device can send the wrong local learning model updates to the aggregation server, and thus delay the convergence of the global federated learning model. Additionally, the global federated learning model might not converge due to wrong local learning model updates. Therefore, there is a need to address this challenge. To do so, Kang *et al.* proposed a reliable federated learning scheme based on consortium blockchain for mobile networks [94]. The proposed scheme tackles the issue of unreliable updates in federated learning process. A non trusty end-device can send unreliable updates to the aggregation server, and thus degrades the global federated learning accuracy. The authors introduced reputation-based metric for reliable federated learning over mobile networks. Using the reputation-based metric, a worker (i.e., reliable end-devices) selection scheme based on consortium blockchain is proposed. The scheme consists of five steps, task publishment, worker selection, reputation collection, federated learning, and reputation updating. In task publishment, task publishers broadcast task with requirements to mobile devices. Mobile devices fulfilling requirements and willing to join a certain task in turn send joining request to publisher. Next in worker selection scheme, the publisher verifies devices and allow only valid devices participation. The workers with reputation values (computed using subjective logic model) greater than the certain threshold are selected. Then, the federated learning step is performed to yield global federated learning model updates. Finally, the reputation values of end-devices are updated to yield the latest updates after blockchain consensus algorithm is finished. Although the proposed scheme offers protection against malicious users in federated learning, the delay associated with the blockchain consensus algorithm might not be desirable. Additionally, the energy consumption associated with the blockchain consensus algorithm must be minimized. For instance, PoW consumes significantly high energy and suffers from high latency but it offers more decentralization. Therefore, depending on the IoT application and requirements, one must develop blockchain consensus algorithms that will consume low energy (e.g., delegated proof of stake) and offer low-latency (e.g., Byzantine fault tolerance) [102].

The works in [93], [97], [99], and [98] applied federated learning various IoT applications. In [97], a framework, namely, FedHealth, based on federated transfer learning was proposed for smart health-care. The purpose of using transfer learning in FedHealth is to reduce the distribution divergence among various domains. Additionally, FedHealth uses homomorphic encryption to enable secure transfer of learning model updates between the aggregation server and end-users. FedAvg has been used as a federated optimization scheme in FedHealth. Although the global model obtained at cloud is available to all end-users, the personalized model for each user might not perform well. Therefore, to improve the performance of the federated learning model for end-users, the authors used transfer learning to learn personalized models. For validation of the FedHealth framework, the public human activity recognition dataset, namely, UCI smartphone was used. The FedHealth framework considered FedAvg as a federated optimization scheme, and thus, its performance can be further improved using FedProx that better considers heterogeneity among end-users. In [93], Yu and Liu proposed a federated learning-based object detection scheme to overcome the privacy leakage issue of deep learning-based object detection schemes. The authors considered three kinds of data distributions such as IID, non-IID, and extreme non-IID. IID has local datasets of similar distribution, whereas non-IID uses a subset of images of a certain type for the same client. In the case of extreme non-IID, the only object of interest is marked (labeled) in contrast to the non-IID case which involves marking all the objects. The authors used abnormal weight suppression in their proposed federated object detection model to enable trimming of the weights (due to non-IID and extreme non-IID local datasets) that are far from normal weights mean. The federated learning-based object detection model was validated for both IID and non-IID data which revealed degraded performance for non-IID data. Furthermore, for extreme non-IID datasets, performance degradation is the highest.

Ye *et al.* in [99] proposed a selective model aggregation-based federated learning framework for image classification task in vehicular edge computing. The proposed framework consists of a centralized server and vehicular clients. The vehicular clients equipped with various sensors to capture images, train their local learning models for image classification tasks after receiving a request from the centralized server. The locally learned models are then sent to the centralized server for global aggregation. To overcome the limitation of FedAvg that is based on a random selection of clients for the learning process, the authors proposed a selective client selection. Selecting clients with low image quality and poor performance degrades the global federated learning model accuracy. Therefore, we must select clients with better performance. However, determining the client local information (i.e., image quality and computational power) at the centralized is server is not available. Coping with this challenge of information asymmetry, a set of devices with fine quality images is selected using contract theory-based scheme. The authors evaluated their scheme for Belgium TSC and MNIST datasets which showed better performance than FedAvg.

Chen *et al.* in [98] studied augmented reality based on federated learning-enabled mobile edge computing. Specifically, the authors focused on the solution of computation efficiency, low-latency object detection, and classification problems of AR applications. A system model for augmented reality applications with object detection and classification problem formulation was presented. One way to solve the problem is through the use of centralized machine learning. However, it suffers from high communication resources consumption costs

TABLE V
A SUMMARY OF STATE-OF-THE-ART ADVANCES

Reference	Local learning model	Dataset	Data Distribution	Aggregation fashion	Analytical convergence analysis	Primary objective	Limitations
Wang *et al.*, [29]	FNN	Xender's trace [101]	Non-IID	Centralized aggregation	✗	Federated learning was used for training of Deep Q-agents in a resource optimized way.	Edge-AI framework has security and privacy concerns.
Wang *et al.*, [88]	SVM, CNN	MNIST, Fashion MNIST	IID, non-IID	Centralized aggregation	✓	Optimized number of local iterations at network edge.	No convergence analysis for non-convex loss functions
Chen *et al.*, [64]	FNN	MNIST, Synthetic dataset	Non-IID	Centralized aggregation	✓	A joint learning and communication framework for federated learning at edge was proposed.	No convergence analysis for non-convex loss functions
Feraudo *et al.*, [89]	FNN	IoT Botnet identification dataset	IID	Centralized aggregation	✗	A CoLearn framework for federated learning at network edge was proposed.	CoLearn framework has security concerns.
Conway-Jones *et al.*, [95]	SVM	MNIST	IID, non-IID	Centralized aggregation	✗	Emulated federated learning in a resource constrained environment.	Lack of efficient criteria for selecting node as an aggregation server
Nishio *et al.*, [45]	CNN	Fashion MNIST, CIFAR-10	IID, non-IID	Centralized aggregation	✗	A client selection protocol for performance improvement in FedAvg was proposed.	FedCS protocol did not consider transmit power allocation than can further improve performance.
Kang *et al.*, [94]	Not available	MNIST	IID, non-IID	Centralized aggregation	✗	A framework to minimize false local learning model injection was proposed.	High latency due to running blockchain consensus algorithm
Chen *et al.*, [97]	CNN	UCI Smartphone [102]	Non-IID	Centralized aggregation	✗	Federated healthcare framework using transfer learning was proposed.	The framework did not effectively address the issue of data heterogeneity.
Yu *et al.*, [93]	CNN	Pascal VOC 2007, Pascal VOC 2012 [103]	IID, non-IID	Centralized aggregation	✗	Federated learning-based object detection scheme was proposed.	Federated optimization scheme can be modified to improve the performance for extreme non-IID case.
Ye *et al.*, [99]	CNN	MNIST, BelgiumTSC [104]	IID	Centralized aggregation	✗	Federated learning framework for vehicular edge computing was proposed.	The framework has robustness issues in case of failure of a centralized aggregation server.
Chen *et al.*, [98]	CNN	CIFAR-10	IID, non-IID	Centralized averaging	✗	Federated learning-based augmented reality framework was proposed.	FedAvg can be replaced by other federated optimization schemes (e.g., FedProx) to further improve the performance.
Qu *et al.*, [91]	CNN	Fashion-MNIST, CIFAR-10	IID	Distributed aggregations	✗	A robust architecture, namely, FL-Block was proposed.	High latency and energy consumption associated with the framework
Savazzi *et al.*, [96]	CNN, 2-NN	Real time industrial IoT setup data	Non-IID	Distributed aggregations	✗	A framework using cooperation among devices to enable federated learning without using centralized aggregation server.	The framework did not consider resource and power allocation that can further improve its performance.
Zhao *et al.*, [90]	MLP	MNIST	IID, non-IID	Hierarchical aggregation	✗	A hierarchical architecture for federated learning using for radio access was proposed.	No theoretical convergence analysis for their proposed scheme
Wang *et al.*, [105]	Not available	CIFAR-10	IID, non-IID	Hierarchical aggregation	✓	Hierarchical federated learning framework was proposed.	A criterion for grouping of devices based on wireless parameters is missing.
Liu *et al.*, [31]	CNN	MNIST, CIFAR-10	IID, non-IID	Hierarchical aggregation	✓	Hierarchical edge-cloud-based federated learning framework was proposed.	Robustness concerns due a centralized global aggregation server

associated with transferring the data from end-devices to the centralized server for training. Moreover, the end-devices of augmented reality-based applications might not want to share their data with the centralized server due to privacy concerns. To solve the formulated problem, the authors proposed a framework based on mobile edge computing enabled by

TABLE VI
PRIMARY CONTRIBUTION AREA, KEY CONTRIBUTIONS, AND EVALUATION OF THE RECENT ADVANCES

Primary contribution area	Reference	Key contributions	Evaluation parameters				
			P1	P2	P3	P4	P5
Edge AI	Wang *et al.*, [88]	• A control algorithm is proposed to minimize learning loss under resource budget constraints. • Performance of the proposed algorithm was evaluated using real time datasets. • A theoretical convergence bound was derived for federated learning at network edge. • The proposed control algorithm learns learning model characteristics, system dynamics, and data distribution.	✗	✗	✗	✗	✗
	Wang *et al.*, [29]	• An EdgeAI framework was proposed. • Intelligent computation offloading and edge caching were considered. • Used double deep Q-learning networks. • Federated learning was considered for reducing the transmission overhead during the training process.	✗	✓	✗	✗	✗
	Nishio *et al.*, [45]	• Proposed a client selection protocol using edge server. • The edge server manages the resources for communication between clients and edge server. • The proposed FedCS mostly has attained accuracy within less time for CIFAR-10 and Fashion-MNIST datasets.	✗	✓	✗	✗	✓
	Feraudo *et al.*, [89]	• A federated learning architecture for edge networks called as CoLearn was proposed. • To enable more flexibility in their architecture, the user policy server was considered that enables network administrator to enforce rules other than defined by manufacturer.	✓	✗	✗	✗	✗
	Zhao *et al.*, *et al.* [90]	• A federated learning-enabled intelligent fog radio access networks framework was proposed. • Considered hierarchical federated learning that offered two stages of learning model aggregation such as local at fog node and global at cloud. • Neural network compression and learning model parameters compression are considered. • Key enabling technologies for enabling federated learning-enabled intelligent fog radio access networks were presented.	✓	✓	✗	✗	✗
	Qu *et al.*, [91]	• Proposed a blockchain-enabled federated learning (FL-Block) for fog computing scenarios to offers robustness against the poisoning attacks. • FL-Block used decentralized fashion for aggregation of local learning model rather than centralized server to offer robustness. • Additionally, decentralized privacy scheme was proposed for FL-Block.	✓	✗	✗	✓	✗
Smart object detection	Yu *et al.*, [93]	• Proposed a smart object detection scheme. • Considered three types of data distribution such as IID. Non-IID, and extreme IID for analysis. • Abnormal weights suppression was used to trim the weights far from average normal weights during training.	✗	✗	✗	✗	✗
Augmented/Virtual reality	Chen *et al.*, [98]	• A system model for augmented reality applications with object detection and classification problem formulation was presented. • To solve the formulated problem, the authors proposed a framework based on mobile edge computing enabled by federated learning.	✗	✗	✗	✗	✗

federated learning. Finally, the authors validated their proposal for image classification tasks using CIFAR-10 datasets for non-IID and IID settings.

B. Advances Based on Distributed Aggregation

The works in [91] and [96] considered federated learning based on a distributed aggregation of end-devices local

TABLE VII
PRIMARY CONTRIBUTION AREA, KEY CONTRIBUTIONS, AND EVALUATION OF THE RECENT ADVANCES

Primary contribution area	Reference	Key contributions	Evaluation parameters				
			P1	P2	P3	P4	P5
Mobile networks	Kang *et al.*, [94]	• Proposed a reliable federated learning scheme based on consortium blockchain for mobile networks. • Using the reputation-based metric, a worker (reliable end-devices) selection scheme based on consortium blockchain was proposed. • Several future research directions were provided.	✓	✗	✗	✗	✗
	Conway-Jones *et al.*, [95]	• Federated learning over emulated wide area communications network having features of intermittent, heterogeneous, and dynamic availability of resources was studied. • Intermittent and dynamic behavior was considered which might made the design suitable for use in various practical applications.	✗	✗	✗	✗	✗
	Savazzi *et al.*, [96]	• Presented federated learning scheme that uses cooperation between devices to avoid use of centralized edge/cloud server for global model aggregation. • Two strategies such as consensus-based federated averaging and consensus-based federated averaging with gradients exchange, were proposed to enable effective distributed optimization.	✗	✓	✗	✓	✗
	Chen *et al.*, [64]	• A framework for joint learning and communication for wireless federated learning was presented. • A problem for joint power allocation, resource block allocation, and user selection problem was formulated. • A close-form expression for the expected convergence rate of federated learning was derived to account for the effect of wireless channel on federated learning.	✗	✓	✗	✗	✓
	Wang *et al.*, [101]	• A hierarchical framework for federated learning was presented. • Detailed convergence analysis were presented.	✗	✓	✗	✗	✗
	Liu *et al.*, [31]	• An edge-cloud-hierarchical federated learning framework was presented. • Convergence analysis with simulation results were presented.	✗	✓	✗	✗	✗
Smart health-care	Chen *et al.*, [97]	• A framework namely, FedHealth, based on federated transfer learning was proposed for smart health-care. • Homomorphic encryption was considered for secure transfer of learning model updates between the end-users and aggregation server. • Transfer learning was used by FedHealth framework to enable effective personalized models for end-users.	✓	✗	✗	✗	✗
Vehicular networks	Ye *et al.*, [99]	• Proposed a selective model aggregation-based federated learning framework for image classification task in vehicular edge computing. • To overcome the limitation of FedAvg that is based on random selection of clients for learning process, the authors proposed a selective clients selection based on contract theory.	✗	✗	✗	✗	✓

learning models for various IoT applications. Here, end-devices compute their local learning models and sent the learning model updates to various distributed aggregation servers. The distributed aggregation servers share the local learning models of their associated end-devices and finally, perform global model aggregation. In contrast to the centralized aggregation server-based federated learning, distributed aggregation servers based federated learning do not require

the use of a centralized aggregation server. Qu *et al.* proposed a blockchain-enabled federated learning (FL-Block) for fog computing scenarios to offers robustness against the poisoning attacks [91]. FL-Block uses decentralized fashion for aggregation of local learning models rather than the centralized server to offer robustness. A single centralized server might get malfunctioned or accessed by an unauthorized malicious user. The process of federated learning in FL-Block starts with the local learning model by all the devices. The local learning models are transmitted to the fog servers where cross-verification takes place. Next, the block is generated by the winning miner and block propagation takes place. Finally, global model updates are computed via the aggregation of local learning model parameters at every fog server and sent to all the devices involved in learning. The authors analyzed the latency of the proposed framework. Additionally, a decentralized privacy-preserving scheme is proposed for FL-Block. The authors provided extensive simulation results to validate their proposal. Specifically, they have shown that the proposed scheme performs superior in terms of learning latency for the scenario without malfunctioning of the miners than malfunctioned miners case. Although the FL-Block offers a key feature of robustness, the consensus algorithms used by blockchain typically require high latency to reach consensus. Therefore, novel schemes will be required for federated learning based on blockchain. On the other hand, the end-devices involved in the learning framework of [96] might not be able to access the fog server due to strict communication resource constraints. To cope with this issue, one can use device-to-device communication that can reuse the channel resources already in use by other cellular users [2]. Other than channel resources reuse, cooperation between devices can enable federated learning without the need for a centralized aggregation server. To do so, the work in [96] presented a federated learning scheme that used cooperation between devices to avoid the use of a centralized edge/cloud server for global model aggregation. To effectively enable all the devices in a network to contribute in distributed optimization of the global federated learning model, two strategies are proposed: consensus-based federated averaging and consensus-based federated averaging with gradients exchange. In consensus-based federated averaging, all the devices receive learning model parameters from other nodes which are subsequently used for updating their own models. On the other hand, consensus-based federated averaging with gradient exchange uses additional steps including consensus-based federated averaging to improve convergence speed, but at the cost of extra complexity. Finally, the authors validate their proposal via experimental results for industrial IoT setup. The advantage of the approach considered in [96] is the non-usage of a centralized server and thus, offers more robust federated learning. Furthermore, the proposed scheme can offer scalability by incorporating more nodes in a dense D2D network. Although the proposed scheme offered several advantages, one must perform efficient resource allocation for dense D2D networks. Optimization theory, heuristic schemes, and game theory can be used for resource allocation. Other than resource allocation, one can propose effective power control schemes to further improve the packet error rates, and thus increase the global federated learning model accuracy. This power control scheme will enable users to optimally allocate the transmit power while fulfilling the power constraints of the system.

C. Advances Based on Hierarchical Aggregation

Considering the aforementioned advantages and disadvantages of both centralized aggregation and distributed aggregation-based federated learning, we can consider the hierarchical fashion of aggregation for federated learning. In hierarchical federated learning, few aggregations of local learning models (e.g., at the edge servers) take place prior to central aggregation (e.g., at cloud server). Few works [31], [90], and [101] consider hierarchical federated learning for IoT networks. A framework for federated learning-enabled intelligent fog radio access networks was proposed in [90]. The authors identified the key issues (i.e., high communication cost and security and privacy issues) of a centralized machine learning-enabled fog radio access networks. To overcome these limitations, the federated learning-based framework was proposed. Both traditional federated learning and hierarchical federated learning were considered. In the hierarchical federated learning, two kinds of aggregation take place, one at fog nodes and the other one at a remote cloud. Furthermore, to enable participation of more end-devices in the training process, local model neural network architecture compression and parameter compression are considered. Finally, key enabling technologies for enabling federated learning-enabled intelligent fog radio access networks and open research challenges were presented. Furthermore, the works in [31] and [101] proposed a hierarchical federated learning framework for mobile networks. Their proposals considered local aggregations prior to global aggregation at the cloud. It must be noted here that hierarchical federated learning can be used to improve performance. End-devices can communicate with the edge by reusing the communication resources already in use by other cellular users. However, careful design considerations, such as frequency of aggregations at the edge server before a global aggregation takes place, must be taken. In hierarchical federated learning based on FedAvg, there will be two kinds of weights divergences, such as the edge (i.e., due to client-edge divergence) and cloud (i.e., due to edge-cloud divergence), where aggregations take place. The upper bound on the weights divergences is dependent on the number of edge aggregations and local learning model iterations [101]. For a fixed product of local iterations and the number of edge aggregations, generally increasing the number of edge aggregations will improve the learning performance for non-IID data. One thing must be noted here that the performance of hierarchical federated learning also depends on data distribution and the federated optimization (i.e., FedAvg and FedProx) scheme used. Therefore, properly handling the data heterogeneity using an effective federated optimization scheme and the numbers of local iterations and edge aggregations will cause performance improvement for hierarchical federated learning.

D. Lessons Learned and Recommendations

We have discussed recent advances of federated learning towards enabling IoT networks. We have learned several lessons and future enhancements.

- From [64], we derived that federated learning must use some effective compression technique due to limited communication resources and a massive number of IoT devices. Several compression techniques such as gradient compression and model broadcast compression can be used [28]. The gradient compression minimizes the size of local learning models that are transmitted to the edge/cloud server. Although compression results in size reduction of the bits used to represent the learning model updates, it might result in data loss. Therefore, compression techniques with low loss in data will need to be developed.
- We derived from [96] that there should be robust federated learning scheme for IoT networks. Vanilla federated learning is based on a single centralized aggregation server, which can be easily attacked by a malicious user or it stops working due to physical damage. These issues result in the interruption of the federated learning process. Therefore, we must tackle these issues to enable robust federated learning over IoT networks. To address the robustness issue, a novel design based on distributed aggregation servers will be required.
- We derived from [45] and [99] that heterogeneity and noise-aware federated learning protocols must be proposed. The set of devices involved in federated learning have different computational resource (CPU-cycles/sec) and dataset sizes. Furthermore, the distribution of their datasets is not always identical. Such a heterogeneous nature of end-devices poses significant challenges to the design of a federated optimization algorithm. For instance, FedAvg was developed without more carefully considering devices' heterogeneous system parameters. To cope with this limitation, FedProx was developed to offer a more practical design. FedProx is based on the addition of a scaled proximal term to FedAvg local device loss function. However, the scaling constant might be difficult to adjust for different applications. On the other hand, we must assign communication resources adaptively (i.e., more communication resources to a device with high local model computation time and vice versa) to optimize the overall global federated learning model time. Furthermore, the noise present in the local devices datasets significantly affects the local learning model accuracy. Therefore, novel federated optimization schemes that consider devices heterogeneity and local datasets noise in the devices selection phase will be required.
- From [91], we derived that distributed aggregation-based federated learning for IoT networks must share the learning model updates in a secure and trustful manner. Recently, blockchain-based federated learning has been proposed for secure and trustful sharing of learning model updates between different miners. Although blockchain can provide many features, it has an inherent issue of high latency associated with consensus algorithms. On the other hand, federated learning consists of a number of global iterations (i.e., communication rounds). Therefore, using blockchain during the training of a federated learning model might not be desirable due to the high latency associated with the blockchain consensus algorithm. To cope with the high latency issue, there is a need to develop low complexity consensus algorithms.

IV. Taxonomy of Federated Learning for IoT

Federated learning over wireless networks involves many players, such as end-devices, edge/cloud servers, and miners [26]. These players use wireless channel and core network resources in addition to computing resources for successfully enabling complex interaction between themselves [27]. Moreover, end-devices must be given some incentives to motivate their participation in federated learning process. Keeping in mind the aforementioned facts, we derive a taxonomy (illustrated in Fig. 9) using operation modes based on global aggregation, resources, local learning models, incentive mechanism, federated optimization schemes, end-device design, miners classification, cloud server design, edge collaboration, security, and privacy, as parameters. A detailed discussion with insights about every parameter of the taxonomy will be given later in this section. In Fig. 9, we place various parameters of taxonomy at three layers: end-devices layer, edge layer, and cloud layer, to provide more clear insights about their relationship to federated learning-enabled IoT networks. End-devices design and local learning models are specific to local devices and it will perform local model training for federated learning. Next to local model training, we can do aggregation either at a remote cloud or edge servers. Cloud servers at the cloud layer can be implemented using container-based design or virtual machine-based design. To perform aggregation at the edge layer, we can use simply an edge server (edge-assisted aggregation) or miners implemented at the edge. The use of miners for blockchain-based federated learning will be explained in more detail later in this section. Moreover, we can use collaboration among edge servers for federated learning global model computation (more details will be given in Section IV-F). On the other hand, incentive mechanism design, security, and federated learning optimization/learning schemes involve end-devices, edge servers, and cloud servers. Therefore, they are positioned by considering all the three layers, such as the devices layer, edge layer, and cloud layer. Next, we describe various parameters of taxonomy for federated learning over IoT networks in detail.

A. Security and Privacy

Federated learning over IoT networks have security and privacy concerns. Although federated learning was introduced to enable users' privacy, it has still privacy leakage concerns.[2]

[2]In this sub-section, we use the keyword "privacy leakage" to end-devices sensitive information leakage due to the information inferring capability of a malicious user or aggregation server using their learning model updates. On the other hand, accessing end-devices due to weak authentication schemes and altering the end-devices are considered as a security concern.

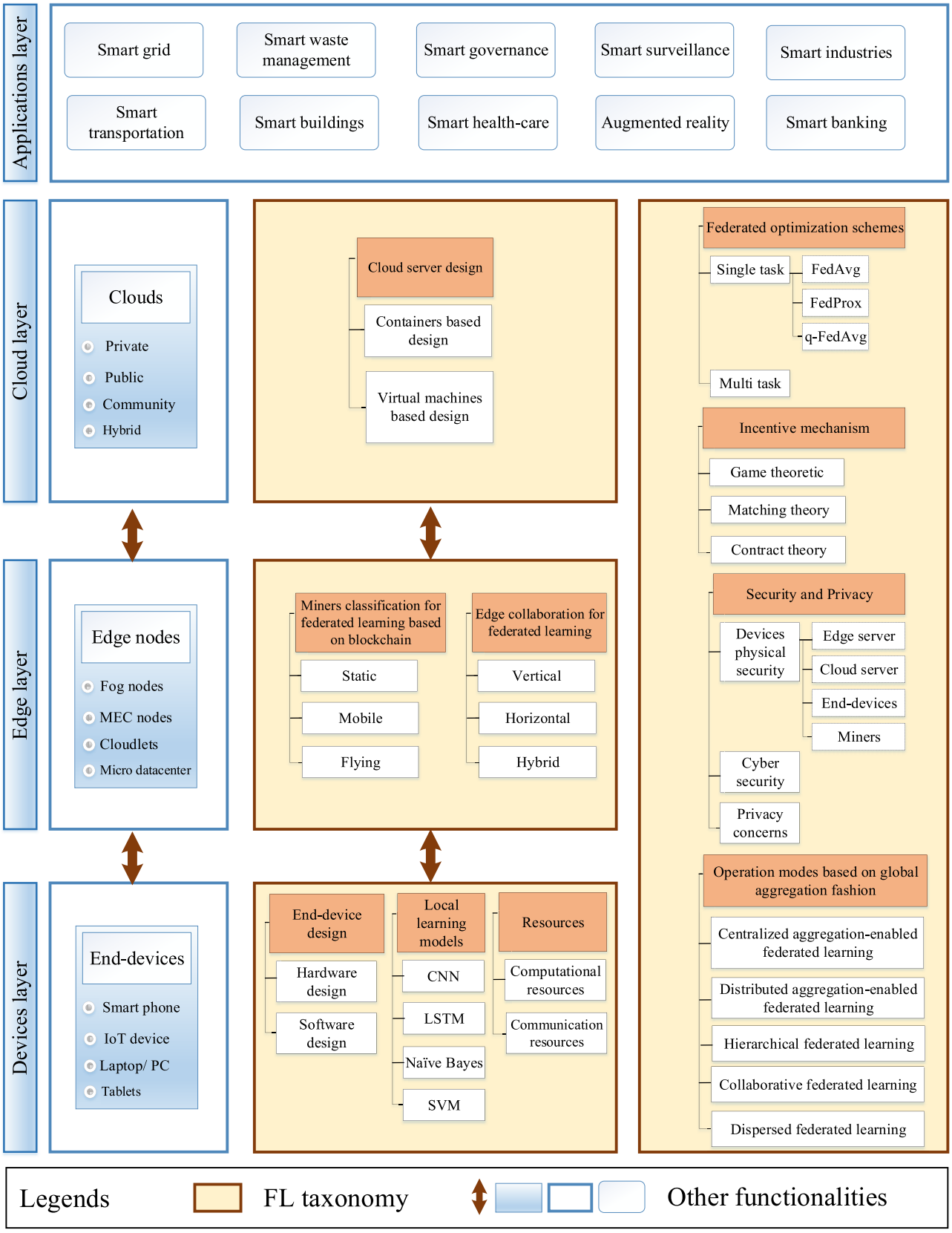


Fig. 9. Taxonomy of federated learning for IoT networks.

A malicious end-device and aggregation server can infer end-devices private information [33], [103]. To address this issue, a differential privacy scheme can be used that is based on the addition of noise to local learning model parameters before sending it to the global aggregation server [26], [104]–[108]. In local differential privacy, each end-device performs differential

private transformation by adding noise which reduces the accuracy of the federated learning model. Noise for local differential privacy can be added via various ways, such as the Gaussian mechanism and the Laplace mechanism. Laplace mechanism adds noise of Laplace distribution [109]. However, adding a Laplace noise directly to the local learning models might cause significant performance degradation especially for larger values of the noise [110], [111]. To cope with this limitation, one can use differential privacy based on an exponential mechanism. An exponential mechanism is based on mapping the local learning model updates to values within a range with a probability that follows an exponential distribution. Although exponential mechanism generally performs better than Laplace mechanism, it might suffer from performance degradation [112]. Another way to improve the performance of local differential privacy is to use a Gaussian mechanism that adds noise according to a zero-mean isotropic Gaussian distribution [112]. The addition of Gaussian noise has the main advantage of being additive in nature. Adding a Gaussian noise to Gaussian distributed local learning model weights will result in another Gaussian distribution whose analysis is simple. In [107], the authors showed that generally adding more noise to local learning model parameters can increase the privacy preservation capability, but at the cost of loss in accuracy and long convergence time. Therefore, we must make a tradeoff between the convergence and privacy preservation level. Another way to avoid prolonging the convergence time would be to use secure multiparty computation-based schemes, achieved by the encoding of local learning model updates. However, this approach will cause more communication overhead and might not be fully effective against the privacy leakage attacks. Therefore, to resolve these issues, one can use a hybrid approach that combines both local differential privacy via the addition of minimum possible noise and secure multiparty computation-based scheme via low-overhead encryption [106].

In addition to privacy leakage, federated learning for IoT networks has a few security concerns. Generally, security in federated learning for wireless networks can be divided into two main categories: devices physical security and cybersecurity. Devices' physical security refers to restricting the physical access to the end-devices [2]. Enabling end-devices, servers, and miners with physical security is challenging. On other hand, cybersecurity refers to network security, information security, and applications security [113], [114]. To ensure the integrity of data stored at end-devices, one must use effective and light-weight authentication schemes due to the fact that a malicious user can physically access the end-devices with fewer efforts [115]–[117]. Similarly, the edge servers have distributed nature and enabling its physical security is very challenging. To enable its security, one must develop new light-weight authentication schemes to provide access to only registered and trustworthy users [118], [119]. The main reason for the development of light-weight authentication schemes for federated learning-enabled IoT networks is due to power constraints of the most IoT devices. In contrast to the edge, the cloud has centralized nature and can be easily attacked by a malicious user [120]. Specifically, the cloud-based federated learning has serious security concerns. A malicious user can easily access the centralized cloud server and introduce error in the global federated learning model. Therefore, similar to edge, we must use effective authentication schemes for cloud security [121]–[123]. It is to be noted here that, the authentication scheme used at the cloud can be of higher complexity than that of edge server due to presence of more computational capacity at the cloud. Other than end-devices, edge servers, and cloud security, federated learning is susceptible to malicious user attacks during the exchange of learning model updates between end-devices and the aggregation server. To address this issue, one can use an effective encryption scheme [106]. It must be noted here that using an effective authentication scheme can preserve privacy as well. A malicious user or aggregation server can not easily infer the end-devices sensitive information from their encrypted local learning model updates. Generally, complexity of an encryption algorithm increases with performance. Therefore, we must make a trade-off between performance and communication overhead and complexity.

B. Resources

Enabling efficient federated learning over an IoT-network requires optimization of both computational and communication resources, as shown in Fig. 10 [27], [124], [125]. The computational resource can be local computational resources of the end-devices or global computational resources of the aggregation server [126]–[129]. To compute the local learning model, the end-device performance (i.e., local learning accuracy and computation time) is mainly determined by its dataset size, available energy, and computational resource (CPU-cycles/sec). The computation time of a local learning model for a fixed accuracy strictly depends on the dataset size and the device computational resource (CPU-cycles/sec). The energy consumption of the device having operating frequency f, dataset size D, and I_l local iterations, is given by [46], [130], [131]:

$$E_{\text{local}} = I_l\left(\rho\zeta D f^2\right), \tag{4}$$

where ρ and ζ are the CPU dependent constant parameter and CPU-cycles required to process a single dataset point, respectively. The local learning model computation time is given by:

$$T_{\text{local}} = I_l\left(\frac{\zeta D}{f}\right). \tag{5}$$

From (4) and (5), it is clear that there exists a tradeoff between simultaneously minimizing local learning model computation time and energy consumption. For a fixed local model accuracy and device operating frequency, the dataset size determines the computation time of the local learning model. For fixed dataset points and local model accuracy, the local model computation time can be decreased by increasing the operating frequency of the device. However, energy consumption increases proportionally to the square of the local device operating frequency. Therefore, we must make a trade-off between the end-device

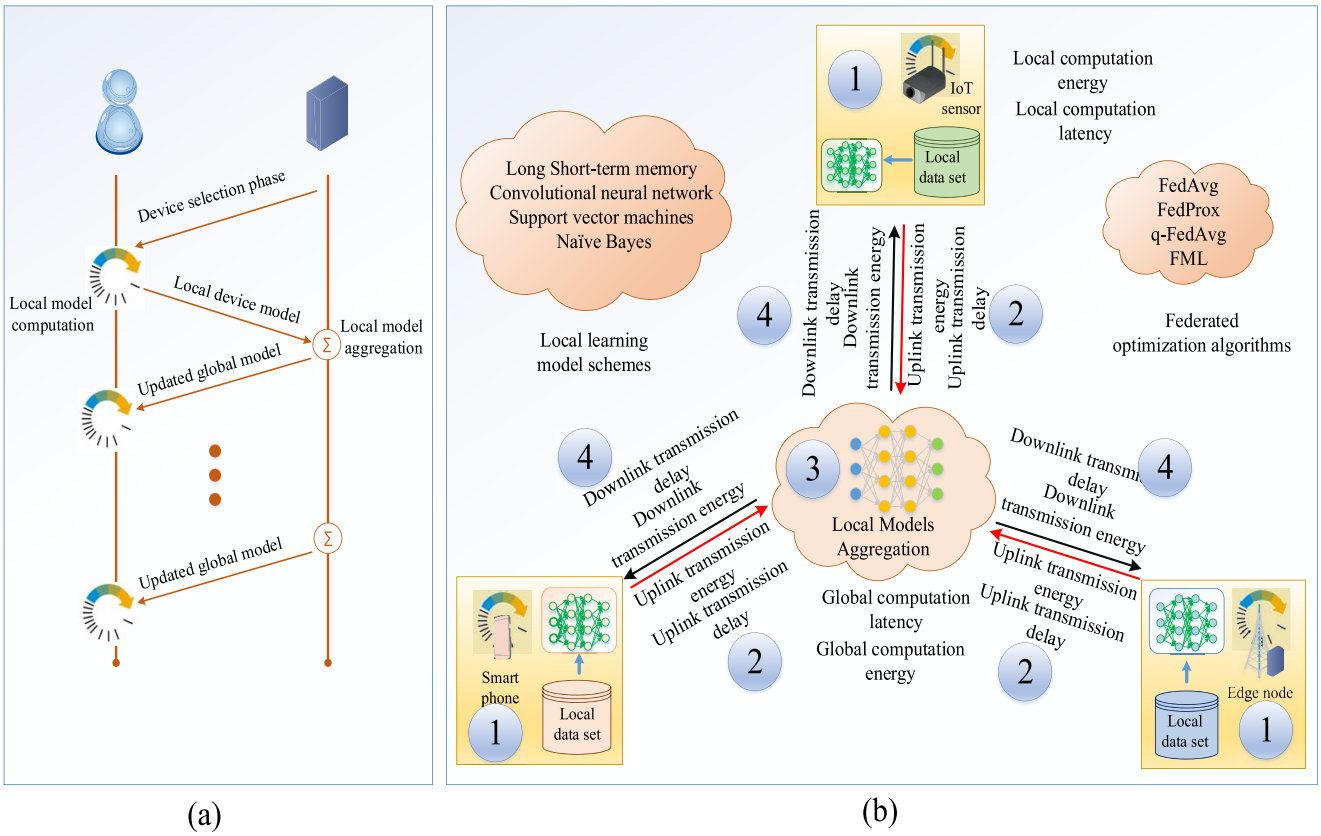


Fig. 10. Federated learning (a) sequence diagram, (b) overview of resources, local learning algorithms, and federated optimization schemes [27].

operating frequency and energy consumption while computing a local learning model. A set of devices involved in federated learning shows significant heterogeneity in terms of computation resource, dataset size, operating frequency, and available energy. For a fixed local model computation time, the end-devices' heterogeneous parameters result in local learning models of variable accuracy. To account for variable local learning model accuracy of end-devices due to heterogeneity, we can use the notion of relative local accuracy θ [132]. Smaller values of relative local accuracy show better local learning model accuracy and vice versa.

After computing the local learning model at every end-device, the learning model parameters are sent to the aggregation server. Depending on the federated learning scheme, the aggregation of the local learning model can be carried out either at a centralized aggregation server or distributed nodes. For a vanilla federated learning, the global model aggregation takes place at the edge server/cloud [27]. The global model updates are sent back to the end-devices. On the other hand, the local learning model updates of all the end-devices are transferred between miner-enabled BSs for a blockchain-based federated learning model [133]. The miners exchange the local learning model parameters of all the devices through distributed ledger in a trustworthy way. After a consensus is reached among the miners, all the miners send the block which contains local learning models of all the devices to their corresponding end-devices where the global model aggregation takes place. The transmission time for transferring of model updates between the BSs and end-devices depends on the radio access scheme and channel variations [49], [134]. For a data rate $R_n, \forall n \in \mathcal{N}$, the time taken during sending of learning model parameters of size ν to the aggregation server can be given by:

$$T_{\text{trans}} = \sum_{n \in \mathcal{N}} \left(\frac{\nu}{R_n} \right). \tag{6}$$

On the other hand, the energy consumed during transmission using transmit power $P_n, \forall n \in \mathcal{N}$ of the learning model parameters from end-devices to the aggregation server is given by:

$$E_{\text{trans}} = \sum_{n \in \mathcal{N}} \left(\frac{\nu P_n}{R_n} \right). \tag{7}$$

From the above discussions regarding computational resource and communication resource consumption, one can define computational and communication costs. The computational cost can be the number of local iterations for local model computation, whereas communication cost can be the number of global communication rounds between the end-devices and aggregation server. For a fixed global federated learning model accuracy, computational and communication costs have contradictory relations with each other. An increase in computational cost decreases the communication cost and vice versa. For a relative local accuracy θ and global federated learning model

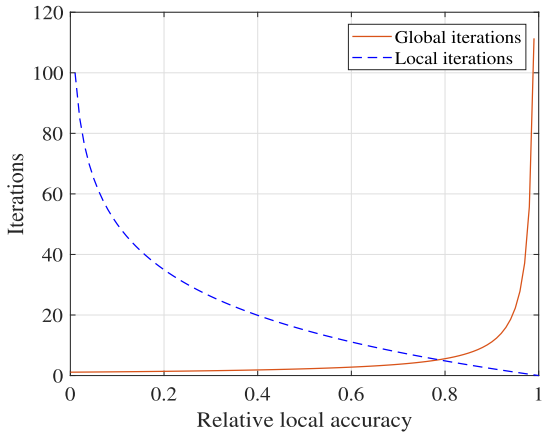


Fig. 11. Global and local iterations vs. relative local accuracy.

accuracy ϵ, the global iterations can be given by [135]:

$$I_g = \frac{\alpha \log\left(\frac{1}{\epsilon}\right)}{1-\theta}, \tag{8}$$

From (8), it is clear that for a fixed global accuracy, more local iterations are required to minimize the relative local accuracy θ (i.e., to maximize the local learning model accuracy). The local iterations can be given by [132]:

$$I_l = \gamma \log\left(\frac{1}{\theta}\right), \tag{9}$$

where γ is constant, which depends on end-device local sub problem and dataset [136]. Fig. 11 shows the variations in global and local iterations vs. relative local accuracy for a fixed global accuracy. To achieve lower values of relative local accuracy, more local iterations and few global iterations are needed for a fixed global accuracy and vice versa.

From the aforementioned discussions, it is evident that there must be some effective mechanism for the selection of local devices operating frequencies for federated learning. The devices' operating frequencies determine the energy consumption and local model computation delay for a fixed number of dataset points and fixed local learning model accuracy [27]. On the other hand, there exist significant variations in the wireless channel and different devices have different transmission delays for sending local learning models to edge/cloud server [137]. Therefore, the joint optimization of energy and latency for federated learning over wireless networks must be considered. In [65], joint optimization of energy and latency for single task federated learning was considered. On the other hand, multitask federated learning has attractive applications in smart industries. Therefore, multitask federated learning schemes that jointly optimize energy and latency will be required.

C. Local Learning Models

In typical federated learning, two types of nodes such as end-devices and centralized edge/cloud servers are involved. First of all, a local learning model is trained at every device using its local dataset. The choice of local learning model strictly depends on the type of IoT task and end-devices computational power with backup energy. For a particular kind of local learning model (e.g., CNN), selecting the number of hidden layers with activation functions determines its performance. Generally, increasing the number of layers improves the local learning performance. However, considering large models that consist of many hidden layers might not always perform well. The reason for this can be the small size of the considered local dataset. It may happen with deep learning models used for local model computation to cause performance degradation with an increase in a number of hidden layers for small local datasets. Therefore, depending on the type of local dataset and task, we should select wisely the number of hidden layers for the local learning model. Additionally, the activation functions must be chosen properly as already mentioned in Section II-C. Various machine learning schemes: Naive Bayes, SVM, CNN, and LSTM can be used for local model training [26]. Summary of local learning models with their IoT application is given in Table VIII [29], [45], [88], [89], [91], [96]–[98]. Additionally, Table VIII provides details about the local learning model architecture and insights about the performance of federated learning for various applications using these local learning models. The CNN models are used in [96]–[98] for smart object detection, mobile networks, healthcare, augmented reality, and vehicular networks. Other works [29], [45], [88], [89], [91] mainly focused on edge intelligence to enable various IoT applications. Typically, end-devices have low computational power and limited backup power. Therefore, running complex deep learning algorithms on end-devices with low computational might not be desirable. To address this issue, one must use some effective local computation complexity reduction approach to minimize the local learning model computation time [28]. We must modify the existing local learning model algorithms to reduce their computational complexity. However, generally reducing the complexity of the deep learning network results in performance degradation. Therefore, we must make a trade-off between local learning model accuracy and complexity.

D. Incentive Mechanism

In order to ensure large-scale adoption of federated learning, there is a need to devise novel incentive mechanisms. Since there are different ways to implement federated learning (e.g., centralized edge/cloud server-based vanilla federated learning and blockchain-based federated learning), there is a need to design different incentive mechanisms that can be appropriately used for the federated learning implementation of interest.

For centralized edge/cloud server-based federated learning, the two players of the federated learning are devices and edge/cloud server, which interact with each other to train the global federated learning model. Numerous ways can be used to model such type of interaction between the devices and edge/cloud server. Several approaches (summary is given in Table IX) have been proposed regarding the design of incentive mechanisms for federated learning [132], [138], [139]. These

TABLE VIII
LOCAL LEARNING MODELS WITH DISCUSSION USED IN VARIOUS FEDERATED LEARNING-ENABLED IOT APPLICATIONS

Reference	Local learning model	Primary contribution	Local model architecture	Discussion
Wang *et al.*, [88]	SVM, CNN	Edge AI	The network has 9 layers with $5\times5\times32$ convolutional, 2×2 MaxPool, local response normalization, $5\times5\times32$ Convolutional, local response normalization, 2×2 MaxPool, $z\times256$ fully connected, 256×10 fully connected, and softmax activation. The other model is squared-SVM.	The authors considered various data distributions for analysis with the aim to find the optimal number of local iterations. Their analysis shows that number of optimal local iterations depends on the type of dataset, data distribution, and local learning model. Meanwhile, their proposed adaptive algorithm generally achieved better performance than traditional federated learning [19] by optimally adjusting local learning model iterations. Although theoretical analysis for convex loss functions are provided, the work did not consider theoretical analysis for non-convex loss functions of many practical scenarios.
Wang *et al.*, [29]	FNN	Intelligent edge caching and computational offloading	Single-layer fully-connected FNN, that has 200 neurons.	The federated learning-based double deep Q-network performed near to centralized double deep Q-network. Meanwhile, transmission cost is also low for federated learning-based double deep Q-network compared to centralized learning-based double deep Q-network. However, its performance is little degraded than the deep Q-network based on centralized training, which may be further improved by considering more complex local learning model (e.g., convolutional neural network with many layers).
Nishio *et al.*, [45]	CNN	Client selection protocol	The model has six convolution layers with 32, 32 , 64, 64, 128, and 128 channels. All the convolutional layer use 3×3 filters. All the layers are are followed by ReLU and batch normalization. After every two layers 2×2 max pooling is used. These convolutional layers are followed by three fully connected layers with 382 and 192 units. The last layer has 10 units with softmax activation.	The proposed client selection protocol showed performance improvement than the random client selection of FedAvg [19] with an increase in deadline that accounts for overall global model computation time using both IID and non-IID settings for fashion MNIST and CIFAR-10 datasets. Although the client selection protocol offered significant performance improvement, it did not consider the scenario of multiple BSs with aggregation server at a remote cloud.
Feraudo *et al.*, [89]	FNN	Framework for intelligent edge	FNN model with two hidden layers having 50 and 30 neurons.	From the analysis of this work, it is revealed that their framework generally results in performance improvement for an increase in number of local iterations. The proposed framework does not effectively tackle the security issues of the IoT devices. One must use effective authentication schemes for IoT devices to avoid injection of false local learning model updates. Injecting false local learning model updates will prolong the global federated learning convergence time.
Qu *et al.*, [91]	CNN	Edge AI	The model has six convolution layers with 32, 32 , 64, 64, 128, and 128 channels. All the convolutional layer use 3×3 filters. All the layers are are followed by ReLU and batch normalization. After every two layers 2×2 max pooling is used. These convolutional layers are followed by three fully connected layers with 382 and 192 units. The last layer has 10 units with softmax activation.	A block-FL framework using blockchain was proposed to avoid data poisoning. For both CIFAR-10 and fashion MNIST, the authors evaluated the performance of proposed scheme by increasing the round time for global model computation. Generally, increasing the number of round time causes an increase in global federated learning accuracy. Although FL-block can offer us with the robust operation, it has additional latency due to blockchain consensus algorithm.
Savazzi *et al.*, [96]	CNN, FNN	Mobile networks	The 2NN model uses fully connected layer of 32 hidden nodes (dimension 512×32) followed by a ReLU layer and a second fully connected layer of dimension $32\times C$.The second CNN model consists of a convolutional layer ($8\times16\times1$) followed by max pooling (5×5) and a fully connected layer of dimension $168\times C$.	A fully decentralized federated learning approach was proposed to minimize global communication rounds and improve convergence. Two schemes such as cooperative federated learning based FedAvg (CFA) and cooperative federated learning based FedAvg with gradient exchange (CFA-GE) were proposed. CFA-GE has shown less validation loss than CFA. Additionally, increasing the number of devices in cooperation also improves performance due to the fact that more devices participate in learning process. Although the proposed scheme can offer several advantages, it will result in additional complexity for performing cooperation between the devices.
Chen *et al.*, [97]	CNN	Healthcare	The model has 2 convolutional layers, 2 pooling layers, and 3 fully connected layers. A convolution size of 1×9 was used.	A federated transfer learning framework was proposed for end-devices with their own local datasets and a public dataset at cloud. The convolutional layers and max-pooling layers are kept frozen (not updated) in model transfer at end-devices due to low-level features extraction characteristic of these layers regarding the end-user activity. Using transfer learning in such as fashion at end-devices results in significant performance improvement. Their proposal out performed support vector machines and k-nearest neighbors algorithm. Furthermore, the proposed framework can be extended with incremental learning for more personalized healthcare.
Chen *et al.*, [98]	CNN	Augmented reality	The network has a convolutional layer (i.e., $3\times6\times5$), maxpool layer (i.e., 2×2), convolutional layer (i.e., $6\times16\times5$), and followed by a fully connected network. The activation function used was ReLU.	This work considered a convolutional neural network model and CIFAR-10 dataset. From the analysis, it is revealed that an increase in local iterations will cause a proportional performance improvement. However, the increase will not be more for higher number of local iterations (i.e., for 15 and 20). Simple averaging in the proposed scheme can be replaced by weighted aggregation scheme that will take into account the wireless fairness issues.

approaches can be classified into game theory, contract theory, and auctions theory. In [132], a crowdsourcing framework was proposed for federated learning to enable a high-quality global model with wireless communication efficiency. A two-stage Stackelberg game-based incentive mechanism was adopted to jointly maximize the utility of the aggregation server and the

TABLE IX
INCENTIVE MECHANISM FOR FEDERATED LEARNING

Reference	Incentive model	Motivation	Device utility/ bid	Aggregation server utility
Pandey *et al.*, [132]	Stackelberg game	To enable a high-quality global federated learning model with wireless communication efficiency.	Difference between reward (i.e., $/accuracy level) and cost (i.e., communication and computation cost)	Concave function of $\log(1/global\ accuracy)$
Le *et al.*, [138]	Auction theory	To minimize social cost which denotes true cost of end-devices bids.	Difference between reward and cost.	Minimizes cost of bids.
Kang*et al.*, [139]	Contract theory	To motivate participation of reliable end-devices in federated learning.	Difference between reward and energy consumption.	Maximize the profit that is given by the difference between total time for global iteration and reward given to end-devices.

end-devices. The utility of the aggregation server was modeled as a concave function of $\log(1/global\ model\ accuracy)$, whereas end-device utility was modeled as a difference between reward (i.e., $/accuracy level) and cost (i.e., communication and computation cost). Le *et al.* presented an auction-based incentive mechanism for federated learning over cellular networks [138]. Every end-device submits a set of bids to the BS. The bid consists of claimed cost and resources (i.e., local accuracy level, CPU cycle frequency, transmission power, and sub-channel bundle). The authors formulated a social cost minimization problem representing the true cost of user bids. To deal with NP-hard social cost minimization problem, randomized auctions based mechanism was proposed. Next, the BS determines the set of all winner profiles. Finally, the winning end-devices participate in the federated learning process. On the other hand, [139] proposed an incentive mechanism for reliable federated learning. The authors defined a reputation-based metric to determine the trustworthiness and reliability of end-devices participating in the federated learning process. Then, a reputation based end-device selection scheme using a multi-weight subjective model was proposed. Moreover, blockchain was considered for secure end-devices reputation management. Finally, they proposed an incentive mechanism based on contract theory to motivate end-devices participation in federated learning.

On the other hand, blockchain-based federated learning consists of sets of devices and miners [126], [133]. First, the devices compute their local learning models and then send the learning model parameters to their associated miners. The miner verifies and exchanges the learning model parameters with other miners and run a consensus algorithm. Once the miners complete the consensus algorithm, a block consisting of verified local learning models is added to the blockchain. The end-device associated with the miners download the block containing the verified local learning models of different devices. Finally, the global federated learning model aggregation is performed at the end-devices. This process of federated learning offers robustness and security but suffers from additional overhead compared to the traditional federated learning approach. Designing an incentive algorithm for such kind of federated learning scheme involves end-devices and miners. The reward of the end-devices must be based on their dataset size and local learning model accuracy. The miners will provide the reward to the end-devices, whereas miners will receive their reward from the blockchain service providers responsible for federated learning model management.

E. Miners Classification for Federated Learning Based on Blockchain

Federated learning based on blockchain uses miners for secure and trustful exchange of learning model parameters [91], [133], [140]. The miners can be either static (i.e., BS) [133], mobile (i.e., autonomous cars) [141], and flying (unmanned aerial vehicles (UAV)) [142]. There are two main challenges in using miners for blockchain-based federated learning, such as wireless for miners and blockchain consensus design for miners. Wireless for miners deals with the wireless communication design for efficient communication, whereas blockchain consensus for miners deals with the efficient consensus algorithm design. The primary purpose of miners is to enable a trustful exchange of local learning models as shown in Fig. 12 [133]. First of all, local learning models are computed for all devices and sent to their corresponding miners. The miners verify the trustfulness of the local learning model to avoid injection of wrong local learning models from a malicious user. Following cross verification, a block of a distributed ledger is generated. The block has two parts body and header. Body stores the local learning model updates and header contains control information, such as pointer to the previous block, output of the consensus algorithm, and block generation rate. Every miner has its block that stores local learning models of its associated devices. Next, every miner generates a hash value repeatedly until their hash value becomes less than the target value. After the miner succeeds in computing the hash value using proof of work (PoW), its block is considered as a candidate block which is transmitted to all other miners. All other miners that receive the newly generated block from the winning miner should stop computing their PoWs. It must be noted that the other miners may broadcast their candidate block after computing a hash value before receiving a candidate block from the other winning

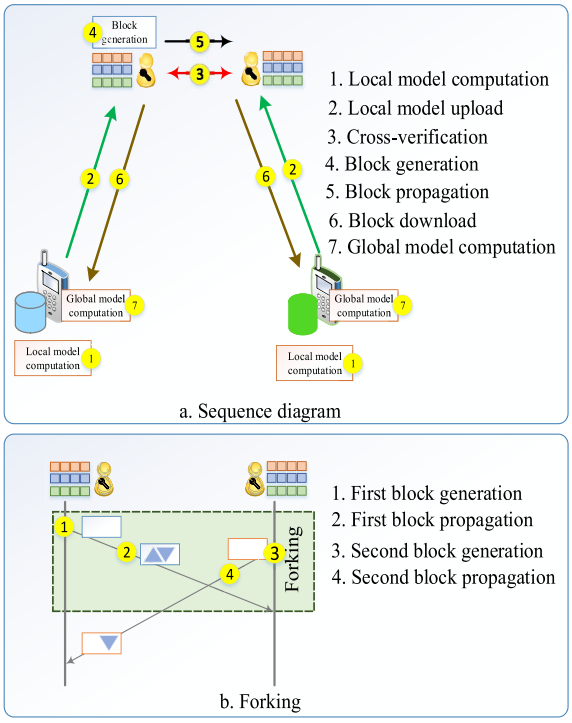


Fig. 12. Blockchain-based federated learning: (a) sequence diagram, and (b) forking effect [133].

miner, and thus there will be two candidate blocks within a blockchain network. This will cause some of the miners to add this second candidate block, and thus will result in global federated learning model error. This event is called forking. After block propagation, the block is downloaded by all the devices and global aggregation is performed. Blockchain-based federated learning avoids the use of a centralized server, and thus offers robustness but at the cost of computational complexity associated with blockchain consensus algorithm. Additionally, more communication resources will be consumed compared to traditional federated learning. Other than these issues, it is necessary to tackle the issue of forking. One way can be to minimize the block generation rate and block transmission delay. Another possible solution can be sending of candidate block by a winning miner with a certain probability. This will reduce the change of forking event, but at the cost of block generation delay. Additionally, setting this probability will be scenario dependent, such as the number of miners and consensus algorithm used. Therefore, we must take proper attention to the design of blockchain-based federated learning.

For BS-based miners, the association of end-devices has lower complexity than mobile and flying miners. Moreover, BS-based miners can communicate with each other using high-speed optical fiber backhaul links. Using fast back-haul links for transferring blocks among miners as discussed earlier will minimize the forking effect. For flying miners based on UAVs, the forking effect will be more prominent due to the large propagation delay between the miners. Furthermore, the mobility of the flying miners will add further complications to the design due to the fact the miner might get out of the communication range of the other miners. There is a need to optimize radio access network resources for BS-based

TABLE X
COMPARISON OF MINERS USED FOR BLOCKCHAIN-BASED FEDERATED LEARNING

Miner type	Available computational power	Forking probability	Block propagation delay
Static miner (edge server-based BS)	Highest	Lowest	Lowest
Flying miner (UAVs)	Low	High	Highest
Mobile miner (autonomous car)	High	Low	Low

miners to minimize the blockchain-based federated learning convergence time. On the other hand, mobile miners require more careful design consideration than static BS-based miners. Mobile miners can communicate with each other via fast backhaul or directly using wireless links. Comparison of various miners that can be used for blockchain-based federated learning are given in Table X. For mobile miners' communication via a wireless channel, there is a need to carefully design resource allocation schemes for carrying out the transfer of learning model updates between miners. Similar to mobile miners, the UAV-based flying miners require a careful design for federated learning. For UAV-based miners, we need to address two kinds of resource allocation problems such as (a) communication of end-devices with UAVs and (b) communication between UAVs. Furthermore, the three-dimensional motion of UAVs imposes more design complexity [143]–[146].

F. Edge Collaboration

Training of a cloud-based federated learning model for a massive number of IoT devices results in significant overhead of communication resources [27]. To cope with this issue, one can consider collaboration between edge servers and a cloud. In [31], a client-edge-cloud hierarchical federated learning framework has been proposed. The framework resolves high communication resources consumption issue associated with the training of cloud-based federated learning models for a massive number of users via collaboration between edge servers and a cloud server. The client-edge-cloud hierarchical federated learning framework performs two levels of aggregations at edge servers and a cloud. Access to the edge servers is easier and requires fewer communication resources than a remote cloud. Therefore, performing edge based-aggregations and then cloud-based aggregation will offer efficient usage of communication resources than the federated learning scheme based on communication between end-devices and cloud. Edge collaboration can be (a) horizontal, (b) vertical, and (c) hybrid (as shown in Fig. 13), depending on the nature of application and available communication resources.

Although federated learning based on collaboration between edge servers and edge-cloud servers can improve the performance in terms of accuracy, it will face weight divergence issues. For collaboration among edge servers and a cloud, there will be two kinds of weights divergences, such

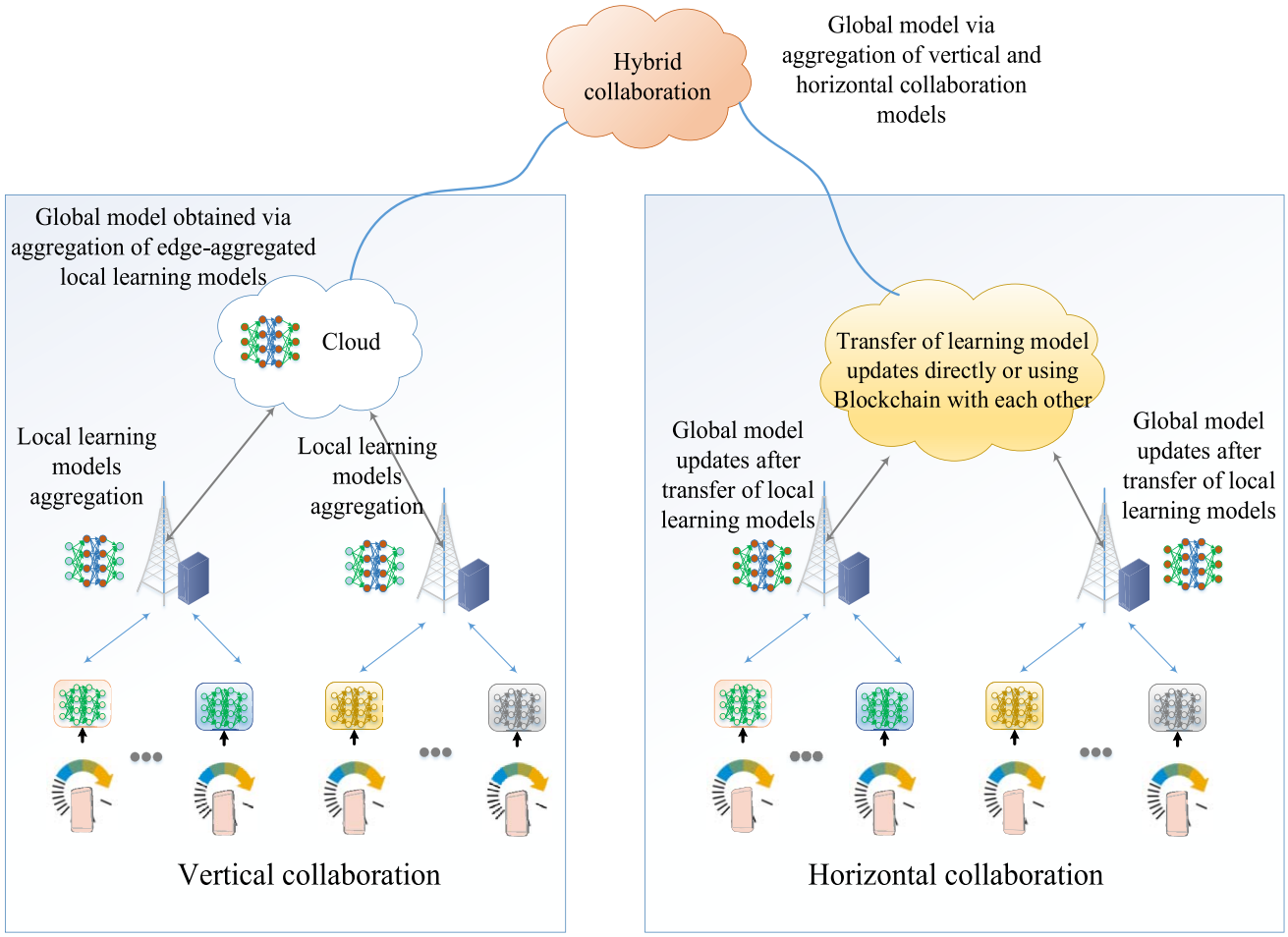


Fig. 13. Edge collaboration categories.

as (a) between end-devices and edge server and (b) between edge servers and cloud. These weight divergences depend on the federated learning scheme used and data heterogeneity. For FedAvg using non-IID data and a fixed product of local iterations and edge aggregations prior to global aggregation, increasing the number of edge aggregations will improve performance [101]. According [101], the reason for this weights divergence is more for a higher number of local iterations compared to edge aggregations for a fixed product of local iterations and edge aggregations prior to cloud aggregation. It must be noted here that we should perform extensive analysis for federated learning based on hybrid collaboration as shown in Fig. 13. Additionally, using a federated optimization scheme different than FedAvg might result in different observations than those of [134]. Therefore, there is a need to properly adjust the federated learning parameters and select an optimization scheme for federated learning based on collaboration.

G. Cloud Server Design

To design federated learning based on the cloud, there is a need to properly plan the design of cloud-based aggregation servers. The cloud-based server will coordinate with devices ranging from hundreds to millions [147]. Additionally, the size of collected updates ranges from kilobytes to tens of megabytes. Therefore, to successfully carry out the federated learning process in an economical way, there is a need to optimally use cloud resources. To truly realize cloud as a dynamic micro data center, there is a need to evolve both hardware and software. Other than hardware evolution, we must effectively isolate functionality from the hardware. Micro-services concept can be used in such a type of isolation. The key requirement of micro-services is a similar environment at all run-time locations. To run micro-services, there can be two ways, i.e., containers-based design [148]–[150] and virtual machine-based design [151], [152]. Containers can be preferably used compared to virtual machines due to their low overhead [2]. Therefore, we can use containerization and hardware evolution for economical cloud server design to enable federated learning for millions of devices.

H. Federated Optimization Schemes

The process of federated learning consists of local model computation by end-devices, which are then sent to the aggregation server. Finally, after global aggregation, the global model parameters are sent back to end-devices. Such a process continues in an iterative fashion until convergence. The goal of federated learning to minimize global loss function that is application dependent. To minimize the global loss function, we need federated optimization schemes. Mainly there

are two types of federated optimization schemes depending on the number of tasks: single task federated optimization and multi-task federated optimization. In single task federated learning, the global federated learning model is trained only for a single task. Several works considered the training of the global federated learning model for a single task [19], [63]. On the other hand, multi-task federated learning involves the training of multiple models for different tasks [153].

FedAvg and FedProx were proposed for training of a single-task federated learning models. FedAvg was based on simple averaging of local learning models at the aggregation server. However, there exist statistical and system heterogeneity among the end-devices. Therefore, simple averaging in FedAvg might not work well. To cope with this challenge, FedProx was proposed [63]. The difference in computation steps between FedProx and FedAvg lies in the local device objective function. FedProx uses an additional proximal term for minimizing the impact of heterogeneity of end-device on the global federated learning model. Although FedProx was developed to account for statistical and system heterogeneity, it faces some practical implementations challenges [40]. The weighted proximal term in FedProx is difficult to choose for various applications. Furthermore, the answer to the question, that can a FedProx provably improve the convergence rate is not clear [28]. A scheme q-FedAvg was proposed to cope with fairness issues due to system heterogeneity (i.e., local model accuracy and wireless resources) [84]. The q-FedAvg modifies the objective function of FedAvg by assigning the end-devices with poor performance higher weights than better-performing end-devices. In q-FedAvg, the devices with higher values of local loss functions are given more importance using large values of q. Specifically, the amount of fairness is determined by the values of q. In another work [88], a scheme offering a trade-off between the local model update and global model aggregation with the aim of reducing the loss function under resource budget constraints was proposed. In [63], [84], [88], all the schemes considered a single task global federated learning model. To offer multi-task learning, federated multi-task learning (FML) was proposed in [153]. Additionally, the FML scheme considered joint optimization of computational and communication resources while fulfilling the fault tolerance constraint.

The iterative exchange of learning model parameters in the federated optimization schemes via wireless channels requires careful design. It will be necessary to develop a novel federated optimization algorithm that considers significantly the effect of packet errors introduced due to channel uncertainties on the global federated learning model. We must define some threshold regarding the packet error rate to determine whether the devices local learning model parameters should be considered or not in the global federated learning model computation. The devices the higher packet error rates must not be considered in training of the global federated learning model computation. One of possible way to improve the packet error rate is the use of channel coding schemes (e.g., turbo codes, convolutional codes, and linear block codes). Different channel coding schemes have different performance and overhead with complexity. Generally, turbo codes outperforms linear block codes but at the cost of communication overhead. Another feasible way can be the use of URLLC codes, such as BCH codes. Therefore, we must make a trade-off while selecting a channel coding scheme for federated learning.

I. Operation Modes Based on Global Aggregation Fashion

We can categorize federated learning into five categories depending on the fashion of global aggregation as centralized aggregation-enabled federated learning, distributed aggregation-enabled federated learning, collaborative federated learning, hierarchical federated learning, and dispersed federated learning [19], [32], [40]. In centralized federated learning, global aggregation takes place only at the centralized edge/cloud server, whereas decentralized aggregation-enabled federated learning avoids the use of a centralized aggregation by performing aggregations at distributed servers. However, hierarchical federated learning involves aggregation of local learning models (e.g., at edge servers) prior to global aggregation at the cloud. In distributed aggregation-enabled federated learning, end-devices are associated with multiple aggregation servers at BSs. The BSs then share the received local learning models from their associated devices with other devices. Finally, aggregation takes place at BSs in a distributed manner. This mode of federated learning can offer robustness but at the cost of additional communication resources associated with sharing of local learning model updates between BSs. Moreover, it will add delay due to sharing of local learning models between BSs. On the other hand, hierarchical federated learning combines both distributed and centralized aggregation-enabled federated learning. Several aggregations of local learning models take place at edge, which is followed by sending the edge aggregated models to the cloud for global aggregation. Hierarchical federated learning offers advantage of using low cost communication for connecting end-devices with the edge servers. At the network edge, one can use easily reuse the already occupied frequency bands by other cellular users while protecting them by keep interference level below the maximum allowed limit.

For scenarios that have massive number of IoT devices and constrained communication resource, it will be difficult for devices to participate in the federated learning process for the aforementioned federated learning schemes. To address this challenge, one can use collaborative federated learning [32]. The collaborative federated learning was introduced mainly for ensuring participation of devices that are unable to access the centralized BS due to communication resources constraints as shown in Fig. 14. In collaborative federated learning, the devices with insufficient communication resources send their model to other nearby devices with sufficient communication resources. The receiving devices perform aggregation of other devices' local learning models, with their own models and send the aggregated model parameters to the centralized BS. Some of the devices send their own local learning models directly to the centralized BS. The centralized BS finally performs global aggregation and sends back the global model parameters to the end-devices. Although collaborative federated learning offers benefits of enabling more devices to

participate in the federated learning process, there is a need for successful communication between devices for sharing the learning model updates. The primary challenge in enabling collaborative federated learning lies in the network formation that will define the communication among devices for local aggregations as shown in Fig. 14. We can use two types of operation for collaborative federated learning, such as synchronous and asynchronous. In a synchronous fashion, the aggregation server must receive a learning model (i.e., local learning models and locally aggregated models) within a maximum allowed time prior to aggregation. To do so, resource-constrained devices must send their local learning models to nearby devices according to the network formation topology within the allowed time. Next, after local aggregation at receiving devices, the locally aggregated models are sent to the centralized aggregation server. It must be noted that the time from local learning model computation to global aggregation in synchronous collaborative federated learning must be less than or equal to the maximum threshold. On the other hand, asynchronous collaborative federated learning has more freedom of computing locally aggregated models. Here, the resource-constrained devices can send their local learning models to nearby devices where local aggregation takes place. If the locally aggregated models are received by the centralized aggregation server later than the some fixed time, then it will consider the local aggregation model in the next round rather than discarding that happens in synchronous collaborative federated learning. Although asynchronous collaborative federated learning allows freedom of allowing more devices in the learning process, it will offer design challenges. Performance degradation in asynchronous collaborative federated learning might occur due to less number of participating end-devices in one global round. Additionally, asynchronous collaborative federated learning might not converge fast for extreme non-IID conditions with most of the end-devices having a distinct class of data. To resolve this challenge, one can use synchronous collaborative federated learning by using fewer local iterations to reduce the local model training time so as to reduce the overall time consumed (i.e., local training and local aggregations along with transmission delay) before the centralized server aggregation.

The aforementioned centralized aggregation-enabled federated learning, hierarchical, and collaborative federated learning faces a robustness challenge. Additionally, these schemes will suffer from privacy leakage due to the end-devices sensitive information inferring capability of a malicious aggregation server/ end-devices. One can enable differential privacy via addition of artificial noise to the learning model parameters prior to sending them to the aggregation server [26]. However, simultaneously achieving convergence time and high privacy level for federated learning is generally not possible. We must make a tradeoff between the federated learning convergence time and privacy level. Additionally, enabling federated learning with privacy preservation schemes adds extra complexity, which might not be desirable. To cope with the aforementioned issues of federated learning over IoT networks, in [40], the novel concept of *dispersed federated learning* was proposed.

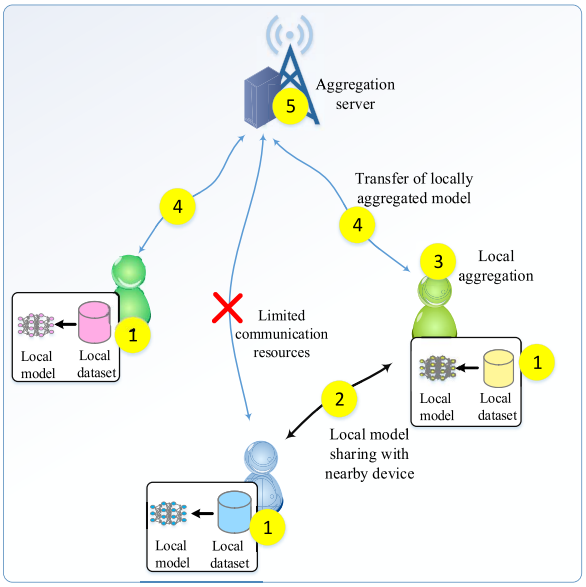


Fig. 14. Collaborative federated learning [32].

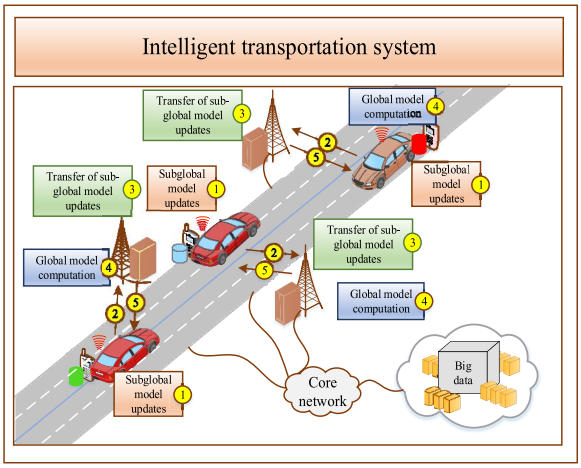


Fig. 15. Intelligent transportation use case of dispersed federated learning.

In dispersed federated learning, sub-global federated learning models are computed within different groups, in a fashion similar to that in federated learning. Then, the sub-global models are transferred between different groups. Next, sub-global models are aggregated to yield the global federated learning model. This process of dispersed federated learning takes place in an iterative manner until desirable global federated learning accuracy is achieved. For dispersed federated learning, we can use reuse communication resources within groups used for sub-global model computation to enable communication efficient operation. Additionally, there is a need to efficiently allocate resources for further performance enhancement. In contrast to federated learning global aggregation server, a global aggregation server used to aggregate sub-global models can not easily infer the end-devices private information, and thus offer enhanced privacy preservation. However, within group used for sub-global computation, the

TABLE XI
COMPARISON OF FEDERATED LEARNING SCHEMES BASED DIFFERENT AGGREGATION FASHIONS

Federated learning scheme	**Robustness**	**Communication resources consumption**	**Implementation complexity**	**Global model computation latency**
	Robustness deals with the successful operation of federated learning scheme in case of failure of aggregation server due to physical damage or security attack.	This parameter refers to the use of communication for federated learning global model computation.	This parameter deals with the implementation complexity of the algorithm used for computing a global federated learning model.	Global model computation latency is the overall latency in computing global federated learning model for one global round.
Centralized aggregation-based federated learning	Lowest	Low (for edge-based federated learning)	Lowest	Low
Distributed aggregation-based federated learning	High	Highest (for blockchain-based federated learning using wireless miners)	Highest (for blockchain-based federated learning using wireless miners)	High
Collaborative federated learning	Low	Low	High (for mobile nodes)	Low
Hierarchical federated learning	Low	Low	Low (edge and cloud based)	Low
Dispersed federated learning	High	Low	Low (edge servers based)	Low

* One sub-global and one-edge aggregation are considered for dispersed federated learning and hierarchical federated learning used for comparison, respectively.

sub-global aggregation server can infer the end-devices private information. To add more privacy preservation to dispersed federated learning, there is a need to develop a novel privacy preservation schemes within groups of dispersed federated learning.

Dispersed federated learning can be easily adopted in many practical IoT applications. An intelligent transportation system suffers from frequent addition of data updates which motivated use of federated learning in contrast to centralized machine learning. For instance, according to statistics, autonomous driving cars generate $4,000$ gigaoctet everyday [30]. Therefore, transferring the whole data for training is not a feasible solution due to high communication overhead. Instead, we can use federated learning that is based on sending of only learning model updates [154]. However, autonomous driving cars suffer from high mobility which causes frequent handoffs. An IoT-based smart device inside an autonomous driving car might not be able to maintain seamless connectivity with the RSU during the training process for federated learning [155]. To tackle this issue, we can use a dispersed federated learning for autonomous driving cars. Fig. 15 shows an overview with a sequence diagram for a smart transportation use case. First of all, a sub-global model is computed within every autonomous car in an iterative manner similar to traditional federated learning model computation. The set of devices within autonomous driving car exchange their local learning modes with the edge computing-based access point. After a particular number of sub-global iterations, the sub-global model updates are sent to the associated RSU by every autonomous driving car. The next step is sharing of the sub-global model updates between different RSUs. The transfer of sub-global model updates is possible using (a) blockchain-based transfer, (b) direct transfer, and (c) light-weight authentication-based transfer. A blockchain-based transfer provides secure transfer and trustful verification of sub-global model updates. However, it suffers from inherent disadvantage of high latency associated with consensus algorithm. The direct transfer of sub-global model updates between different RSUs might suffer from false data injection, and thus prone to error due to malicious attacks. To cope with the aforementioned issues of false data injection and high-latency, we can use light-weight authentication-based scheme for secure transfer of sub-global model updates.

J. End-Device Design for Federated Learning

On-device learning is the key essence of federated learning. Enabling end-devices with local machine learning requires careful design requirements. Every device is characterized by a fixed maximum computational power (CPU-cycles per) and limited backup power. For fixed dataset points, to minimize local learning model computation time, there is a need to run the device at high CPU-cycles/sec. However, device power consumption increases with an increase in operating frequency. Therefore, there is a need to trade-off between the end-devices operating frequencies and power consumption. The key metrics generally considered for performance evaluation of embedded machine learning are cost, throughput/latency, energy consumption, and learning accuracy. For a fixed dataset, the performance of end-devices involved in federated learning depends on both hardware design and software design [27]. On the other hand, for a fixed hardware and software design, the performance matrices are strongly dependent on the used dataset and application.

Various datasets and applications offer a variety of complexities to the embedded machine learning design. For instance, MNIST [19] dataset used for image classification task has lower complexity than GoogLeNet [156] dataset that is used for classification of 1000 image classes. Therefore, there is a need for efficient hardware design with programmability for end-devices to handle various applications offering different complexities. The programmable nature enables the use of generic hardware for different models. The requirement of programmability and high dimensionality causes an increase in data movement and computation power consumption due to more generated data, respectively [157]. The programmability deals with storage and reading the weights, which results in significant energy consumption [158]. Application-specific manycore processors are attractive candidates to enable different designs while fulfilling area, temperature, and energy constraints [159], [160]. However, various challenges exist in the implementation of these systems. First of all, fine-tuning is required for implementing effective run-time resource managers. Second, the use case significantly modifies the design objectives and deployment platform. Third, the evaluation of these designs is expensive and slow. Machine learning can be used to provide highly optimized design. Kim *et al.* [161], proposed machine learning to offer computationally efficient hardware design optimization for manycore systems.

For fixed hardware design, we can search different neural architectures for optimal out of available ones. NAS allows us to find the optimal architecture out of many available architectures for a particular dataset and application [79], [162]–[164]. NAS can be considered to be a subfield of automated machine learning that deals with the complete process from dataset to its preferable machine learning models without knowing expert-level knowledge of machine learning [165]. An overview of a NAS is given in Fig. 16a [166]. Several possible neural architectures can be chain-structured, multi-branch networks, and cell search space, are shown in Fig. 16b. The chain-structured neural networks consist of a sequence of layers with the first layer as input and last layer as output. Each layer receives an input from the preceding layer. In such a type of neural search space, we can adjust the number of layers, operation at each layer (i.e., convolution, depth-wise separable convolutions, or pooling), and hyperparameters (i.e., number of filters and kernel size and strides for a convolutional layer). The hyperparameters are dependent on operation layer, and thus this search space is conditional. On the other hand, multi-branch networks have nodes that can take their inputs as function of previous layers' output as shown in Fig. 16b. The multi-branch networks has more degree of freedom than chain-structured neural networks. To provide more freedom of design, we can use the cell search space approach that involves searching in cells.

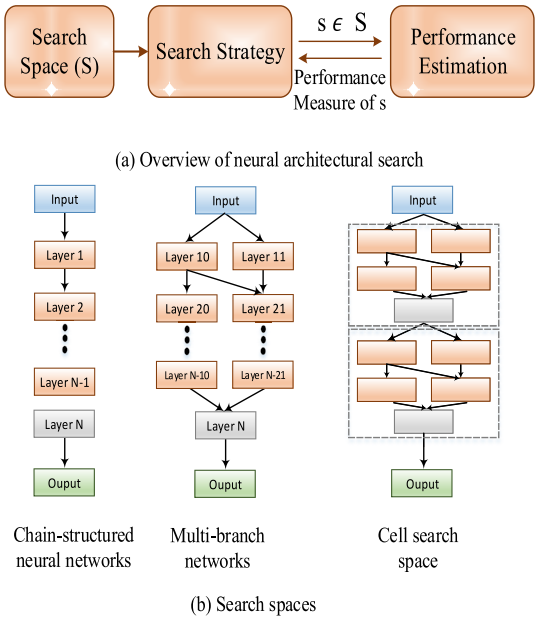


Fig. 16. Neural architecture search: (a) Overview, and (b) Search spaces [165].

K. Lessons Learned and Recommendations

We have provided a taxonomy for federated learning towards enabling IoT-based smart applications. Some of the lessons and future enhancements learned are as follows.

- It is necessary to adaptively adjust local iterations and global iterations strictly depending on the nature of the application. The end-device in an IoT network with high mobility might not have seamless connectivity with the aggregation server all the time. In these scenarios, it is recommended to consider a high number of local iterations. For a fixed global federated learning model accuracy, an increase in the number of local iterations will require less global iterations. Therefore, a higher number of local iterations with few global iterations are desirable in high mobility networks. A practical example such networks is vehicular networks that have high mobility vehicles.
- A massive number of IoT nodes show a significant amount of heterogeneity in terms of local dataset sizes and distributions, available backup power, computation resource, and communication resources. All of these factors significantly affect the performance of federated learning algorithms (i.e., FedAvg, FedProx, etc.) Therefore, to truly benefit from the deployment of federated learning in IoT networks, it is indispensable to design federated learning algorithms that consider these parameters. For instance, some of the end-devices might have poor performance due to limited computation power, noisy dataset, or limited communication resources. Such a type of end-device will have less influence on the global model than the device with better performance. Coping with these kind of issues due to the heterogeneity of devices, we must propose new federated learning algorithms.
- Considering the aforementioned software-aware design and hardware design schemes, it is necessary to consider hardware-software co-design for end-devices involved in federated learning. The hardware-software

co-design allows the concurrent design of both hardware and software. Numerous ways that can be used for hardware-software co-design are high-level synthesis, co-verification-based embedded systems, and virtual prototyping. The virtual prototyping-based design leverages computer-aided engineering, computer-aided design, and computing automated design, to validate the design prior to implementation. A high-level synthesis design enables automated digital hardware using an algorithmic description of the desired system. Multiplexer and wired signal delays are prominent challenges in high-level synthesis-based design. In co-verification-based design, embedded systems are designed using testing and debugging of both hardware and software at the same time. Such a design offers better performance.

- A malicious end-device can send a learning model updates to an aggregation server which can result in prolonging the federated learning convergence time. Therefore, one can propose novel federated learning frameworks that enable trust verification of end-device before using its local learning model updates. One way is through blockchain-based authentication schemes. Another way is through light-weight authentication schemes.
- As federated learning involves iterative exchange of learning model updates between the end-devices and aggregation, efficient communication resources allocation schemes will be required. The two kinds of communication resources used in federated learning are core network and radio access network. For optical fiber-based core networks, there are enough communication resources. On the other hand, the radio access network has communication resources constraints. Therefore, wireless resources optimization problem will need to be solved for federated learning. Depending on the nature of the formulated problem, we can use various approaches. These approaches include heuristic approaches, matching theory-based schemes, and game theory-based schemes.
- FML learning was proposed in [153] to enable learning of multi-tasks whose data are distributed among multiple nodes with federated learning settings (i.e., data and system heterogeneity). To enable federated learning for a different tasks, efficient scheduling over resource-constrained radio access network is required for all tasks. Moreover, resources must be assigned as per the nature of the learning task. For instance, more resources should be assigned to the critical tasks (i.e., industrial control) than ordinary tasks (i.e., keyboard keyword suggestion). Assigning more resources lowers the convergence time for federated learning. Therefore, novel joint scheduling and resource allocation schemes for FML will need to be developed.
- In dispersed federated learning, the communication of the sub-global aggregation server with end-devices is more frequent than with the centralized aggregation server. Therefore, we can use high-performance channel coding schemes (i.e., turbo codes) having high overhead for reliable communication between the sub-global aggregation server and the global aggregation server. It must be noted that the sub-global model results from frequent, iterative aggregation and exchange of various local learning models, therefore, it must be provided with more reliable communication. On the other hand, we can use low-overhead linear block codes or convolutional codes for reliable communication between end-devices and sub-global aggregation server due to their frequent communication.
- There is a need to propose novel communication resource-efficient federated optimization schemes for dispersed federated learning. One can use D2D communication-assisted dispersed federated learning. In D2D communication assisted dispersed federated learning, a set of closely located devices form clusters which is followed by cluster head selection using some attractive criteria (e.g., socially-aware interaction). The devices within a cluster compute their local learning model and exchange it with a cluster head in an iterative manner to yield a sub-global model. The advantage of sub-global model computation lies in the reuse of the already occupied spectrum by the cluster devices. The sub-global model updates from all the cluster heads are sent to the BS where the global model aggregation takes place. Finally, the global model updates are sent back to the cluster heads which disseminate the global model to the devices.
- Mobility management for groups used to compute sub-global models in a dispersed federated learning is of significant importance. A set of devices within a sub-group used for sub-global model computation is desirable to remain within the group until the iterative computation of the sub-global model. Therefore, we must handle the issue of mobility.

V. Open Research Challenges

This section presents various open challenges that exist in enabling federated learning over IoT networks. The challenges presented in our paper are different than those of existing surveys and tutorials related to federated learning for wireless networks, as given in Table XII [26], [29], [37], [38]. We present causes and possible solutions to these challenges. Summary of the open research challenges with their possible solutions for federated learning are given in Table XIII.

A. Sparsification-Enabled Federated Learning

How do we enable federated learning for a massive number of heterogeneous devices under strict wireless resource constraints? Optimal use of limited wireless resources in federated learning for a massive number of heterogeneous devices can be enabled via sparsification, which allows only a small set of devices to send their local learning model parameters to edge/cloud server. The criterion for choosing the set of devices to send their data to the edge/cloud server is a challenging task. One way is to choose the devices with gradients of magnitude greater than a certain threshold. However, computing the

TABLE XII
SUMMARY OF EXISTING SURVEYS AND TUTORIALS OPEN RESEARCH CHALLENGES

Reference	Challenges
Lim *et al.*, [26]	Dropped participants, privacy concerns, unlabeled data, interference among mobile devices, communication security, asynchronous FL, comparisons with other distributed learning methods, further studies on learning convergence, usage of tools to quantify statistical heterogeneity, combined algorithms for communication reduction, cooperative mobile crowd ML, applications of FL.
Li *et al.*, [37]	Extreme communication schemes, communication reduction and pareto frontier, novel models of asynchrony, heterogeneity diagnostics, granular privacy constraints, beyond supervised learning, productionizing federated learning, benchmarks.
Wang *et al.*, [29]	Accelerating AI tasks by edge communication and computing systems, efficiency of In-Edge AI for real-time mobile communication system toward 5G, incentive and business model of In-Edge AI.
Our Tutorial	Sparsification-enabled federated learning, data-heterogeneity-aware-clustering-enabled federated learning, mobility-aware federated learning, homomorphic encryption-enabled federated learning, secure and trustful aggregation-enabled federated learning, interference-aware resource efficient federated learning, interference-aware association for federated learning, quantization-enabled federated learning, adaptive-resource-allocation-enabled federated learning.

optimal value of the threshold for a massive number of heterogeneous devices is difficult. Another criterion for the selection of end-devices can be their local learning model accuracy which is affected by many factors such as clean datasets, local computational power, and backup power. Furthermore, devices with degraded performance over wireless channels result in prolonging the federated learning convergence time due to high packet error rates. To address the above challenges, there is a need for novel effective sparsification enabled federated learning protocols.

B. Data-Heterogeneity-Aware-Clustering-Enabled Federated Learning

How does one enable federated learning for massively distributed devices having heterogeneous features? A massive number of IoT devices exhibit a significant amount of statistical heterogeneity in their datasets. FedAvg was developed to yield a global federated learning model using the assumption of synchronous amount of work at end-devices during each global iteration [19]. However, IoT devices show significant heterogeneity in their datasets and system parameters. To address this issue, FedProx was developed that is based on the addition of a weighted proximal term to the global model of FedAvg [63]. Choosing the weights for FedProx for different applications is challenging. Additionally, it is not clear whether FedProx can provably improve convergence rate [28]. Therefore, there is a need to devise a novel protocol for federated learning to cope with data heterogeneity. One possible solution can be the formation of clusters of end-devices with statistical homogeneity. Within each cluster, a cluster head is selected which acts for aggregation of local learning models. The selection of cluster heads must follow attractive criteria. For instance, the criterion for cluster head selection can be an improvement in the overall throughput of the cluster. Another possible way can be socially-aware clustering. As cluster head has the capability to infer end-devices sensitive information from their local learning model [26], it can be desirable to form cluster head with high social trust nature. Within each cluster, a sub-global model can be computed similarly to traditional federated learning. Next, cluster heads send their sub-global models to global aggregation server to yield a global federated learning model, which is then disseminated to cluster heads. Finally, the cluster heads disseminate the global model updates to devices of their corresponding clusters.

C. Mobility-Aware Federated Learning

How do we enable seamless communication of mobile devices with the centralized edge/cloud server during training phase of federated learning? The set of mobile devices involved in learning must be connected to the edge/cloud server during the training phase. However, mobility of the devices might cause loss in coverage and thus, causes performance degradation of federated learning. Several solutions can be proposed to tackle this issue. One way is modification in the federated learning protocols that should consider the mobility of users in addition to other factors during the devices selection phase. The devices with no mobility or with less mobility than other high mobility nodes must be preferred. This approach seems to be promising in the scenarios where the minimum required number of devices for federated learning have no or limited mobility. However, if the devices have high mobility then it is necessary to predict the mobility of devices during the device selection phase. For this, deep learning-based mobility prediction schemes can be used.

D. Homomorphic Encryption-Enabled Federated Learning

How does one enable secure exchange of federated learning parameters between devices and the cloud/edge server? Although federated learning has been proposed to enable privacy-aware on-device machine learning in a distributed manner, it suffers from security challenges and privacy challenges. A malicious user can access the learning model updates during their transmission between end-devices and the aggregation server. Then, the malicious user abnormally alters the learning parameters. Moreover, the learning parameters can be used by a malicious user to infer end-device sensitive information [26]. Therefore, it is indispensable to ensure secure federated learning over wireless channels. One can use homomorphic encryption for ensuring secure federated learning. In homomorphic encryption-based federated learning, a key-pair is synchronized among all end-devices via a secure channel [56], [167]–[169]. Initially, all the devices encrypts their local learning model updates and send the ciphertexts to a central server. Next, the server sends

TABLE XIII
SUMMARY OF THE RESEARCH CHALLENGES AND THEIR GUIDELINES

Challenges	Taxonomy relevancy	Causes	Guidelines
Sparsification-Enabled Federated Learning	Federated optimization schemes (client selection protocol)	• Significant changes in the in end-devices local accuracy performance due to noisy datasets, local computational power, and local energy. • Wireless resources constraints.	• Gradient-based sparsification scheme. • Selection of devices with sufficient computational, clean datasets, and communication resources.
Mobility-Aware Federated Learning	Federated optimization schemes	• Connectivity loss due to mobility during training phase. • Variations in SINR due to mobility.	• Priority in selection of static and low-mobility devices for training. • Deep learning-based mobility prediction schemes.
Data-Heterogeneity-Aware-Clustering-Enabled Federated Learning	Federated optimization schemes	• Existence of significant statistical heterogeneity among massive number of devices datasets. • Statistical heterogeneity significantly degrades the federated learning convergence performance.	• Clustering based on statistical homogeneity. • Selection of most trustful end-devices acting for aggregation within clusters.
Homomorphic encryption-enabled federated learning	Security and privacy	• Ability of a malicious user to alter learning model parameters during wireless transfer between the aggregation server and end-devices. • Ability of man-in-the-middle to infer end-devices sensitive information from learning model parameters.	• Partially homomorphic encryption. • Somewhat homomorphic encryption. • Fully homomorphic encryption.
Secure and trustful aggregation-enabled federated learning	Security and privacy	• Presence of malicious end-devices. • Wrong local learning model parameters prolong federated learning convergence time.	• Secure aggregation-enabled federated learning scheme. • Consortium blockchain-based trustful verification of end-devices updates.
Interference-aware resource efficient federated learning	Resource optimization	• Communication resources constraints. • Protection of cellular users while reusing frequency by end-devices used in federated learning.	• Game theory-based resource allocation. • Matching game with externalities for resource allocation. • Relaxation-enabled resource allocation scheme.
Interference-aware association for federated learning	Resource optimization	• Dependency of SINR on association. • Proportional effect of association on packet error rate.	• Game theory-based association. • Relaxation-enabled association scheme.
Quantization-enabled federated learning	Resource optimization	• Communication resource constraints. • Massive number of devices are expected in future for federated learning training-enabled IoT applications.	• Gradient-based selective end-devices selection. • Selection of end-devices with better performance.
Adaptive-resource-allocation-enabled federated learning	Resource optimization	• Heterogeneity of computational resources (CPU-cycles/sec), local device energy, and communication resources results in variable local accuracy.	• Adaptive resource allocation (i.e., more wireless resources to devices with high local model computational time) to minimize the overall federated learning convergence time.

back the aggregated result to the end-devices. The advantage of using homomorphic encryption is non-requirement of decryption at the aggregation server. However, generally, homomorphic encryption results in the use of extra computation and communication resources. Homomorphic encryption can be divided into three main categories such as partially homomorphic encryption, somewhat homomorphic encryption, and fully homomorphic encryption [170]. These categories of homomorphic encryption offer freedom of performing mathematical operation depending on the type used.

For instance, partially homomorphic encryption allows only one type of mathematical operation (i.e., addition or multiplication), whereas fully homomorphic encryption allows a large number of different mathematical operations. On the other hand, ciphertext in homomorphic encryption contains a noise which increases proportionally with mathematical operations [171]. Therefore, there is a need to devise a homophobic encryption scheme that accounts for such type of noise.

E. Secure and Trustful Aggregation-Enabled Federated Learning

How do we enable aggregation of learning models at the aggregation server in a secure and trustful way? A malicious user can send the wrong local learning model parameters to the aggregation server to slow the federated learning convergence rate. In some cases, the global federated learning model might not converge due to the presence of a malicious user. On the other hand, it is necessary to verify the end-devices' updates before considering them for global aggregation. Several possible ways can be used to enable secure aggregation. In [172], the authors proposed a secure aggregation protocol for federated learning. The protocol can offer security against $1/3$ of the total devices but at the cost of extra communication resources and complexity. Although the protocol in [172] offers reasonable performance, there is a need to propose novel protocols than can handle malicious users more than $1/3$ of the total devices. In another work, a reputation-based metric was introduced for the trustful participation of end-devices in the federated learning process [94]. Consortium blockchain was then applied to enable efficient reputation management. Although the scheme presented in [94] can offer resilience against adversaries, it has high-latency associated with blockchain. To address the above challenges, there is a need for federated learning protocols that can offer security and trustful verification using low complexity schemes.

F. Interference-Aware Resource Efficient Federated Learning

How do we reuse the uplink communication resources already in use by other cellular users for federated learning while protecting them? To protect cellular users from interference due to end-devices involved in federated learning, we will require resource allocation schemes. There can be two different ways of allocating resource blocks to end-devices. The first one is to assign a single resource block to only one end-device while fulfilling the constraint the maximum allowed interference. Another approach is to assign multiple end-devices to a single resource block to more effectively reuse the resources. For the later case, we can use one-to-one matching, whereas for this case we can use one-to-many matching theory with externalities for resource allocation due to the fact that preference profiles depend both on resource blocks along with other end-devices assigned to the same resource block [173]–[176].

G. Interference-Aware Association for Federated Learning

How does one associate end-devices to the edge servers-enabled small cell BSs (SBSs) to reuse the uplink cellular spectrum while protecting cellular users? For a fixed resource block allocation and fixed transmit power, the end-devices' association has no impact on interference control for cellular users while reusing their frequencies. However, joint transmit power allocation and device association for end-devices can be performed to simultaneously minimize federated learning cost and cellular user interference. Such kind of joint transmit power allocation and the device-association optimization problem is a mixed-integer non-linear programming problem that can be solved sub-optimally in a variety of ways.

H. Quantization-Enabled Federated Learning

How do we enable federated learning for a massive number of devices under communication resource constraints? Although sparsification-enabled federated learning can reduce communication resource consumption, it might not be sufficient alone for a large number of devices and limited communication resources. To address this issue, we can use quantization-enabled federated learning. However, quantization induces errors in global federated learning due to the distortion of the signals. Such kind of errors in the global federated learning model might prolong its convergence time. Therefore, there is a need to make a trade-off between a federated learning convergence time and communication resources consumption during training. Several quantization techniques (i.e., universal vector quantization [177], low-precision quantizer [178], and hyper-sphere quantization [179]) can be used for federated learning over IoT networks.

I. Adaptive-Resource-Allocation-Enabled Federated Learning

How does one perform training of the federated learning model with low convergence time for end-devices with heterogeneous resources? The devices involved in federated learning are heterogeneous in terms of computational resources (CPU-cycles/sec), local device energy, and communication resources. These heterogeneous parameters have a significant effect on the performance of federated learning. For instance, computation time of local device learning models strictly depends on local model accuracy, local device operating frequency, and dataset size for the local learning model algorithm. On the other hand, synchronous federated learning is based on the computation of all the devices' local models within fixed time. However, the local computation time for devices with heterogeneous features shows significant variations in local learning model accuracy for fixed local model computation time. Therefore, these challenges of heterogeneity in the devices must be resolved. One way to reduce the impact of heterogeneity on the performance of federated learning is to allocate more computational resources to users with larger local training dataset sizes for achieving certain accuracy within the allowed time and vice versa. Another approach is to assign more communication resources to the end-devices with poor performance due to a lack of computational resources. Allocating resources adaptively can

reduce the global federated learning time, and thus offer fast convergence.

VI. Conclusion and Future Prospects

In this paper, we have discussed federated learning for IoT networks. Specifically, we have presented state-of-art-advances of federated learning towards enabling smart IoT applications. We have provided a taxonomy using various parameters. Furthermore, we have identified several important issues (i.e., robustness, privacy, and high-communication resources consumption) with the existing federated learning schemes based on a centralized aggregation server and presented guidelines to solve them. Finally, we have presented several open research challenges along with their causes and guidelines. We have concluded that dispersed federated learning will serve as a key federated learning scheme for future IoT applications using a massive number of end-devices.

Although IoT networks enabled by federated learning and fifth-generation (5G) of cellular networks, can offer a wide variety of smart applications such as intelligent transportation, smart industry, smart health-care, among others. However, there are several Internet of Everything (IoE) applications that seem difficult to be fulfilled by 5G. These IoE applications are haptics, flying vehicles, extended reality, brain-computer interfaces, and autonomous connected vehicles. The requirements of these applications seem difficult to be fulfilled by 5G [1]. Therefore, there is a need to propose a new generation of wireless systems, dubbed as sixth-generation (6G) wireless systems. Machine learning will be considered to be an integral part of 6G [1], [6], [180], [181]. Additionally, privacy preservation will be given primary importance in 6G [182]. Therefore, dispersed federated learning with homomorphic encryption can be a promising candidate for the future 6G-enabled IoE applications. In dispersed federated learning, homomorphic encryption will give privacy preservation against a malicious sub-global aggregation server, whereas sub-global aggregations will give privacy preservation against malicious global aggregation server.

References

[1] L. U. Khan, I. Yaqoob, M. Imran, Z. Han, and C. S. Hong, "6G wireless systems: A vision, architectural elements, and future directions," *IEEE Access*, vol. 8, pp. 147029–147044, 2020.

[2] L. U. Khan, I. Yaqoob, N. H. Tran, S. Kazmi, T. N. Dang, and C. S. Hong, "Edge computing enabled smart cities: A comprehensive survey," *IEEE Internet Things J.*, vol. 7, no. 10, pp. 10200–10232, Oct. 2020.

[3] R. Sánchez-Corcuera *et al.*, "Smart cities survey: Technologies, application domains and challenges for the cities of the future," *Int. J. Distrib. Sens. Netw.*, vol. 15, no. 6, Jun. 2019, Art. no. 1550147719853984.

[4] C. Perera, Y. Qin, J. C. Estrella, S. Reiff-Marganiec, and A. V. Vasilakos, "Fog computing for sustainable smart cities: A survey," *ACM Comput. Surveys*, vol. 50, no. 3, pp. 1–43, Jun. 2017.

[5] L. U. Khan, W. Saad, D. Niyato, Z. Han, and C. S. Hong, "Digital-twin-enabled 6G: Vision, architectural trends, and future directions," 2021. [Online]. Available: arXiv:2102.12169.

[6] W. Saad, M. Bennis, and M. Chen, "A vision of 6G wireless systems: Applications, trends, technologies, and open research problems," *IEEE Netw.*, vol. 34, no. 3, pp. 134–142, May/Jun. 2020.

[7] A. Al-Fuqaha, M. Guizani, M. Mohammadi, M. Aledhari, and M. Ayyash, "Internet of Things: A survey on enabling technologies, protocols, and applications," *IEEE Commun. Surveys Tuts.*, vol. 17, no. 4, pp. 2347–2376, 4th Quart., 2015.

[8] C. Perera, A. Zaslavsky, P. Christen, and D. Georgakopoulos, "Context aware computing for the Internet of Things: A survey," *IEEE Commun. Surveys Tuts.*, vol. 16, no. 1, pp. 414–454, 1st Quart., 2014.

[9] *Number of Connected IoT Devices Will Surge to 125 Billion by 2030*. Accessed: Oct. Jun. 14, 2020. [Online]. Available: https://sst.semiconductor-digest.com/2017/10/number-of-connected-iot-devices-will-surge-to-125-billion-by-2030/

[10] *The Growth in Connected IoT Devices Is Expected to Generate 79.4zb of Data in 2025, According to a New IDC Forecast*. Accessed: Oct. Jun. 14, 2020. [Online]. Available: https://www.idc.com/getdoc.jsp?containerId=prUS45213219

[11] K. Qi and C. Yang, "Popularity prediction with federated learning for proactive caching at wireless edge," in *Proc. IEEE Wireless Commun. Netw. Conf.*, May 2020, pp. 1–6.

[12] I. Raicu, I. Foster, A. Szalay, and G. Turcu, "Astroportal: A science gateway for large-scale astronomy data analysis," in *Proc. Teragrid Conf.*, Feb. 2006, pp. 12–15.

[13] P. A. Bernstein and E. Newcomer, *Principles of Transaction Processing*. Burlington, MA, USA: Morgan Kaufmann, 2009.

[14] J. Verbraeken, M. Wolting, J. Katzy, J. Kloppenburg, T. Verbelen, and J. S. Rellermeyer, "A survey on distributed machine learning," 2019. [Online]. Available: arXiv:1912.09789.

[15] D. Alistarh. *Distributed Machine Learning: A Brief Overview*. Accessed: Mar. 2020. [Online]. Available: https://www.podc.org/data/podc2018/podc2018-tutorial-alistarh.pdf

[16] K. Zhang, S. Alqahtani, and M. Demirbas, "A comparison of distributed machine learning platforms," in *Proc. Int. Conf. Comput. Commun. Netw.*, Jul./Aug. 2017, pp. 1–9.

[17] J. Qiu, Q. Wu, G. Ding, Y. Xu, and S. Feng, "A survey of machine learning for big data processing," *EURASIP J. Adv. Signal Process.*, vol. 2016, no. 1, p. 67, May 2016.

[18] D. Peteiro-Barral and B. Guijarro-Berdiñas, "A survey of methods for distributed machine learning," *Progr. Artif. Intell.*, vol. 2, no. 1, pp. 1–11, Nov. 2013.

[19] B. McMahan, E. Moore, D. Ramage, S. Hampson, and B. A. y Arcas, "Communication-efficient learning of deep networks from decentralized data," in *Proc. Artif. Intell. Stat.*, Lauderdale, FL, USA, Apr. 2017, pp. 1273–1282.

[20] C. Yang, Q. Wang, M. Xu, S. Wang, K. Bian, and X. Liu, "Heterogeneity-aware federated learning," 2020. [Online]. Available: arXiv:2006.06983.

[21] Z. Chai *et al.*, "Towards taming the resource and data heterogeneity in federated learning," in *Proc. Conf. Oper. Mach. Learn. (OpML)*, May 2019, pp. 19–21.

[22] J. Konečnỳ, H. B. McMahan, F. X. Yu, P. Richtárik, A. T. Suresh, and D. Bacon, "Federated learning: Strategies for improving communication efficiency," 2016. [Online]. Available: arXiv:1610.05492.

[23] I. F. Akyildiz, W. Su, Y. Sankarasubramaniam, and E. Cayirci, "Wireless sensor networks: A survey," *Comput. Netw.*, vol. 38, no. 4, pp. 393–422, 2002.

[24] J. C. Jiang, B. Kantarci, S. Oktug, and T. Soyata, "Federated learning in smart city sensing: Challenges and opportunities," *Sensors*, vol. 20, no. 21, p. 6230, 2020.

[25] V. D. Nguyen, S. K. Sharma, T. X. Vu, S. Chatzinotas, and B. Ottersten, "Efficient federated learning algorithm for resource allocation in wireless IoT networks," *IEEE Internet Things J.*, vol. 8, no. 5, pp. 3394–3409, Mar. 2021.

[26] W. Y. B. Lim *et al.*, "Federated learning in mobile edge networks: A comprehensive survey," *IEEE Commun. Surveys Tuts.*, vol. 22, no. 3, pp. 2031–2063, 3rd Quart., 2020.

[27] L. U. Khan *et al.*, "Federated learning for edge networks: Resource optimization and incentive mechanism," 2019. [Online]. Available: arXiv:1911.05642.

[28] P. Kairouz *et al.*, "Advances and open problems in federated learning," 2019. [Online]. Available: arXiv:1912.04977.

[29] X. Wang, Y. Han, C. Wang, Q. Zhao, X. Chen, and M. Chen, "In-edge AI: Intelligentizing mobile edge computing, caching and communication by federated learning," *IEEE Netw.*, vol. 33, no. 5, pp. 156–165, Sep./Oct. 2019.

[30] *Federated Learning, A Step Closer Towards Confidential AI*. Accessed: Jan. 24, 2020. [Online]. Available: https://medium.com/frstvc/tagged/thoughts

[31] L. Liu, J. Zhang, S. Song, and K. B. Letaief, "Client-edge-cloud hierarchical federated learning," in *Proc. IEEE Int. Conf. Commun. (ICC)*, 2020, pp. 1–6.

[32] M. Chen, H. V. Poor, W. Saad, and S. Cui, "Wireless communications for collaborative federated learning in the Internet of Things," 2020. [Online]. Available: arXiv:2006.02499.

[33] M. Nasr, R. Shokri, and A. Houmansadr, "Comprehensive privacy analysis of deep learning: Passive and active white-box inference attacks against centralized and federated learning," in *Proc. IEEE Symp. Security Privacy*, May 2019, pp. 739–753.

[34] R. C. Geyer, T. Klein, and M. Nabi, "Differentially private federated learning: A client level perspective," 2017. [Online]. Available: arXiv:1712.07557.

[35] *Technology & Media/Artificial Intelligence Market*. Accessed: Feb. 5, 2021. [Online]. Available: https://www.fortunebusinessinsights.com/industry-reports/artificial-intelligence-market-100114

[36] *Hardware & Software IT Services/Internet of Things (IoT) Market*. Accessed: Feb. 5, 2021. [Online]. Available: https://www.fortunebusinessinsights.com/industry-reports/internet-of-things-iot-market-100307

[37] T. Li, A. K. Sahu, A. Talwalkar, and V. Smith, "Federated learning: Challenges, methods, and future directions," 2019. [Online]. Available: arXiv:1908.07873.

[38] Q. Li, Z. Wen, and B. He, "Federated learning systems: Vision, hype and reality for data privacy and protection," 2019. [Online]. Available: arXiv:1907.09693.

[39] L. Lyu, H. Yu, and Q. Yang, "Threats to federated learning: A survey," 2020. [Online]. Available: arXiv:2003.02133.

[40] L. U. Khan, W. Saad, Z. Han, and C. S. Hong, "Dispersed federated learning: Vision, taxonomy, and future directions," 2020. [Online]. Available: arXiv:2008.05189.

[41] S. A. Rahman, H. Tout, H. Ould-Slimane, A. Mourad, C. Talhi, and M. Guizani, "A survey on federated learning: The journey from centralized to distributed on-site learning and beyond," *IEEE Internet Things J.*, vol. 8, no. 7, pp. 5476–5497, Apr. 2021.

[42] M. Aledhari, R. Razzak, R. M. Parizi, and F. Saeed, "Federated learning: A survey on enabling technologies, protocols, and applications," *IEEE Access*, vol. 8, pp. 140699–140725, 2020.

[43] F. Hussain, R. Hussain, S. A. Hassan, and E. Hossain, "Machine learning in iot security: Current solutions and future challenges," *IEEE Commun. Surveys Tuts.*, vol. 22, no. 3, pp. 1686–1721, 3rd Quart., 2020.

[44] N. Waheed, X. He, M. Ikram, M. Usman, S. S. Hashmi, and M. Usman, "Security and privacy in IoT using machine learning and blockchain: Threats and countermeasures," *ACM Comput. Surveys*, vol. 53, no. 6, pp. 1–37, 2020.

[45] T. Nishio and R. Yonetani, "Client selection for federated learning with heterogeneous resources in mobile edge," in *Proc. IEEE Int. Conf. Commun.*, Shanghai, China, 2019, pp. 1–7.

[46] L. U. Khan, M. Alsenwi, Z. Han, and C. S. Hong, "Self organizing federated learning over wireless networks: A socially aware clustering approach," in *Proc. Int. Conf. Inf. Netw. (ICOIN)*, Barcelona, Spain, Mar. 2020, pp. 453–458.

[47] L. U. Khan, M. Alsenwi, I. Yaqoob, M. Imran, Z. Han, and C. S. Hong, "Resource optimized federated learning-enabled cognitive Internet of Things for smart industries," *IEEE Access*, vol. 8, pp. 168854–168864, 2020

[48] L. U. Khan, U. Majeed, and C. S. Hong, "Federated learning for cellular networks: Joint user association and resource allocation," in *Proc. 21st Asia-Pac. Netw. Oper. Manag. Symp. (APNOMS)*, 2020, pp. 405–408.

[49] M. M. Amiri and D. Gündüz, "Federated learning over wireless fading channels," *IEEE Trans. Wireless Commun.*, vol. 19, no. 5, pp. 3546–3557, May 2020.

[50] A. R. Elkordy and A. S. Avestimehr, "Secure aggregation with heterogeneous quantization in federated learning," 2020. [Online]. Available: arXiv:2009.14388.

[51] N. Shlezinger, M. Chen, Y. C. Eldar, H. V. Poor, and S. Cui, "Federated learning with quantization constraints," in *Proc. IEEE Int. Conf. Acoust. Speech Signal Process. (ICASSP)*, Barcelona, Spain, May 2020, pp. 8851–8855.

[52] D. Alistarh, D. Grubic, J. Li, R. Tomioka, and M. Vojnovic, "QSGD: Communication-efficient SGD via gradient quantization and encoding," in *Proc. Adv. Neural Inf. Process. Syst.*, 2017, pp. 1709–1720.

[53] R. Xu, N. Baracaldo, Y. Zhou, A. Anwar, and H. Ludwig, "HybridAlpha: An efficient approach for privacy-preserving federated learning," in *Proc. 12th ACM Workshop Artif. Intell. Security*, 2019, pp. 13–23.

[54] Z. Yang, S. Hu, and K. Chen, "FPGA-based hardware accelerator of homomorphic encryption for efficient federated learning," 2020. [Online]. Available: arXiv:2007.10560.

[55] J. Tran, F. Farokhi, M. Cantoni, and I. Shames, "Implementing homomorphic encryption based secure feedback control," *Control Eng. Pract.*, vol. 97, Apr. 2020, Art. no. 104350.

[56] C. Zhang, S. Li, J. Xia, W. Wang, F. Yan, and Y. Liu, "BatchCrypt: Efficient homomorphic encryption for Cross-silo federated learning," in *Proc. USENIX Annu. Tech. Conf. (USENIX ATC)*, Santa Clara, CA, USA, Jul. 2020, pp. 493–506.

[57] M. Shirvanimoghaddam *et al.*, "Short block-length codes for ultra-reliable low latency communications," *IEEE Commun. Mag.*, vol. 57, no. 2, pp. 130–137, Feb. 2018.

[58] M. Sybis, K. Wesolowski, K. Jayasinghe, V. Venkatasubramanian, and V. Vukadinovic, "Channel coding for ultra-reliable low-latency communication in 5G systems," in *Proc. IEEE 84th Veh. Technol. Conf. (VTC-Fall)*, 2016, pp. 1–5.

[59] V. Vladyslav, K. Volodymyr, Z. Sergei, and U. Anna, "Adaptive turbo codes for safety in wireless Internet of Things," in *Proc. IEEE 9th Int. Conf. Dependable Syst. Services Technol. (DESSERT)*, 2018, pp. 188–193.

[60] X. Bao, C. Su, Y. Xiong, W. Huang, and Y. Hu, "FLChain: A blockchain for auditable federated learning with trust and incentive," in *Proc. 5th Int. Conf. Big Data Comput. Commun. (BIGCOM)*, 2019, pp. 151–159.

[61] G. Hua, L. Zhu, J. Wu, C. Shen, L. Zhou, and Q. Lin, "Blockchain-based federated learning for intelligent control in heavy haul railway," *IEEE Access*, vol. 8, pp. 176830–176839, 2020.

[62] J. Chen, X. Pan, R. Monga, S. Bengio, and R. Jozefowicz, "Revisiting distributed synchronous SGD," 2016. [Online]. Available: arXiv:1604.00981.

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

[64] M. Chen, Z. Yang, W. Saad, C. Yin, H. V. Poor, and S. Cui, "A joint learning and communications framework for federated learning over wireless networks," 2019. [Online]. Available: arXiv:1909.07972.

[65] N. H. Tran, W. Bao, A. Zomaya, and C. S. Hong, "Federated learning over wireless networks: Optimization model design and analysis," in *Proc. IEEE Conf. Comput. Commun.*, Paris, France, Apr./May 2019, pp. 1387–1395.

[66] H.-S. Ham, H.-H. Kim, M.-S. Kim, and M.-J. Choi, "Linear SVM-based android malware detection for reliable IoT services," *J. Appl. Math.*, vol. 2014, no. 4, 2014, Art. no. 594501.

[67] S. S. Yadav and S. M. Jadhav, "Deep convolutional neural network based medical image classification for disease diagnosis," *J. Big Data*, vol. 6, no. 1, p. 113, 2019.

[68] A. Ndikumana, N. H. Tran, K. T. Kim, C. S. Hong, and D. H. Kim, "Deep learning based caching for self-driving cars in multi-access edge computing," *IEEE Trans. Intell. Transp. Syst.*, vol. 22, no. 5, pp. 2862–2877, May 2021.

[69] X. Du, H. Zhang, H. Van Nguyen, and Z. Han, "Stacked LSTM deep learning model for traffic prediction in vehicle-to-vehicle communication," in *Proc. IEEE 86th Veh. Technol. Conf. (VTC-Fall)*, 2017, pp. 1–5.

[70] K. Thar, N. H. Tran, T. Z. Oo, and C. S. Hong, "DeepMEC: Mobile edge caching using deep learning," *IEEE Access*, vol. 6, pp. 78260–78275, 2018.

[71] S. K. Singh, M. M. Salim, J. Cha, Y. Pan, and J. H. Park, "Machine learning-based network sub-slicing framework in a sustainable 5G environment," *Sustainability*, vol. 12, no. 15, p. 6250, 2020.

[72] E. Hodo *et al.*, "Threat analysis of IoT networks using artificial neural network intrusion detection system," in *Proc. Int. Symp. Netw. Comput. Commun. (ISNCC)*, 2016, pp. 1–6.

[73] L. Li, Y. Xu, Z. Zhang, J. Yin, W. Chen, and Z. Han, "A prediction-based charging policy and interference mitigation approach in the wireless powered Internet of Things," *IEEE J. Sel. Areas Commun.*, vol. 37, no. 2, pp. 439–451, Feb. 2019.

[74] Y. Liu, Z. Ma, Z. Yan, Z. Wang, X. Liu, and J. Ma, "Privacy-preserving federated k-means for proactive caching in next generation cellular networks," *Inf. Sci.*, vol. 521, pp. 14–31, Jun. 2020.

[75] H. Hewamalage, C. Bergmeir, and K. Bandara, "Recurrent neural networks for time series forecasting: Current status and future directions," 2019. [Online]. Available: arXiv:1909.00590.

[76] C. S. Htwe, Y. M. Thant, and M. M. S. Thwin, "Botnets attack detection using machine learning approach for IoT environment," *J. Phys. Conf. Series*, vol. 1646, no. 1, 2020, Art. no. 012101.

[77] M. Bhatia, S. K. Sood, and A. Manocha, "Fog-inspired smart home environment for domestic animal healthcare," *Comput. Commun.*, vol. 160, pp. 521–533, Jul. 2020.

[78] *7 Types of Activation Functions in Neural Network*. Accessed: Jan. 4, 2021. [Online]. Available: https://www.analyticssteps.com/blogs/7-types-activation-functions-neural-network

[79] S. Xie, H. Zheng, C. Liu, and L. Lin, "SNAS: stochastic neural architecture search," 2018. [Online]. Available: arXiv:1812.09926.

[80] C. He *et al.*, "FedML: A research library and benchmark for federated machine learning," 2020. [Online]. Available: arXiv:2007.13518.

[81] T. Ryffel *et al.*, "A generic framework for privacy preserving deep learning," 2018. [Online]. Available: arXiv:1811.04017.

[82] A. Ingerman and K. Ostrowski. (2019). *Introducing TensorFlow Federated*. [Online]. Available: https://blog.tensorflow.org/2019/03/introducing-tensorflow-federated.html

[83] S. Caldas *et al.*, "LEAF: A benchmark for federated settings," 2018. [Online]. Available: arXiv:1812.01097.

[84] T. Li, M. Sanjabi, A. Beirami, and V. Smith, "Fair resource allocation in federated learning," 2019. [Online]. Available: arXiv:1905.10497.

[85] E. Bagdasaryan, A. Veit, Y. Hua, D. Estrin, and V. Shmatikov, "How to backdoor federated learning," in *Proc. Int. Conf. Artif. Intell. Stat.*, 2020, pp. 2938–2948.

[86] Y. Ma, D. Yu, T. Wu, and H. Wang, "PaddlePaddle: An open-source deep learning platform from industrial practice," *Front. Data Comput.*, vol. 1, no. 1, pp. 105–115, 2019.

[87] *PaddlePaddle/PaddleFL*. Accessed: Jan. 7, 2021. [Online]. Available: https://github.com/PaddlePaddle/PaddleFL

[88] S. Wang *et al.*, "Adaptive federated learning in resource constrained edge computing systems," *IEEE J. Sel. Areas Commun.*, vol. 37, no. 6, pp. 1205–1221, Jun. 2019.

[89] A. Feraudo *et al.*, "CoLearn: Enabling federated learning in MUD-compliant IoT edge networks," in *Proc. 3rd ACM Int. Workshop Edge Syst. Anal. Netw.*, Heraklion, Greece, Apr. 2020, pp. 25–30.

[90] Z. Zhao, C. Feng, H. H. Yang, and X. Luo, "Federated-learning-enabled intelligent fog radio access networks: Fundamental theory, key techniques, and future trends," *IEEE Wireless Commun.*, vol. 27, no. 2, pp. 22–28, Apr. 2020.

[91] Y. Qu *et al.*, "Decentralized privacy using blockchain-enabled federated learning in fog computing," *IEEE Internet Things J.*, vol. 7, no. 6, pp. 5171–5183, Jun. 2020.

[92] B. Liu, L. Wang, and M. Liu, "Lifelong federated reinforcement learning: A learning architecture for navigation in cloud robotic systems," *IEEE Robot. Autom. Lett.*, vol. 4, no. 4, pp. 4555–4562, Oct. 2019.

[93] P. Yu and Y. Liu, "Federated object detection: Optimizing object detection model with federated learning," in *Proc. 3rd Int. Conf. Vis. Image Signal Process.*, Vancouver, BC, Canada, Aug. 2019, pp. 1–6.

[94] J. Kang, Z. Xiong, D. Niyato, Y. Zou, Y. Zhang, and M. Guizani, "Reliable federated learning for mobile networks," *IEEE Wireless Commun.*, vol. 27, no. 2, pp. 72–80, Apr. 2020.

[95] D. Conway-Jones, T. Tuor, S. Wang, and K. K. Leung, "Demonstration of federated learning in a resource-constrained networked environment," in *Proc. IEEE Int. Conf. Smart Comput.*, Washington, DC, USA, Jun. 2019, pp. 484–486.

[96] S. Savazzi, M. Nicoli, and V. Rampa, "Federated learning with cooperating devices: A consensus approach for massive IoT networks," *IEEE Internet Things J.*, vol. 7, no. 5, pp. 4641–4654, May 2020.

[97] Y. Chen, X. Qin, J. Wang, C. Yu, and W. Gao, "FedHealth: A federated transfer learning framework for wearable healthcare," *IEEE Intell. Syst.*, vol. 35, no. 4, pp. 83–93, Jul./Aug. 2020, doi: 10.1109/MIS.2020.2988604.

[98] D. Chen *et al.*, "Federated learning based mobile edge computing for augmented reality applications," in *Proc. Int. Conf. Comput. Netw. Commun.*, Big Island, HI, USA, Feb. 2020, pp. 767–773.

[99] D. Ye, R. Yu, M. Pan, and Z. Han, "Federated learning in vehicular edge computing: A selective model aggregation approach," *IEEE Access*, vol. 8, pp. 23920–23935, 2020.

[100] Z. Du, C. Wu, T. Yoshinaga, K.-L. A. Yau, Y. Ji, and J. Li, "Federated learning for vehicular Internet of Things: Recent advances and open issues," *IEEE Open J. Comput. Soc.*, vol. 1, pp. 45–61, 2020, doi: 10.1109/OJCS.2020.2992630.

[101] J. Wang, S. Wang, R.-R. Chen, and M. Ji, "Local averaging helps: Hierarchical federated learning and convergence analysis," 2020. [Online]. Available: arXiv:2010.12998.

[102] U. Majeed, L. U. Khan, I. Yaqoob, S. A. Kazmi, K. Salah, and C. S. Hong, "Blockchain for IoT-based smart cities: Recent advances, requirements, and future challenges," *J. Netw. Comput. Appl.*, vol. 181, May 2021, Art. no. 103007.

[103] A. Gad. *Breaking Privacy in Federated Learning*. Accessed: Mar. 2020. [Online]. Available: https://heartbeat.fritz.ai/breaking-privacy-in-federated-learning-77fa08ccac9a

[104] Y. Zhao *et al.*, "Local differential privacy based federated learning for Internet of Things," 2020. [Online]. Available: arXiv:2004.08856.

[105] Y. Lu, X. Huang, Y. Dai, S. Maharjan, and Y. Zhang, "Differentially private asynchronous federated learning for mobile edge computing in urban informatics," *IEEE Trans. Ind. Informat.*, vol. 16, no. 3, pp. 2134–2143, Mar. 2020.

[106] S. Truex *et al.*, "A hybrid approach to privacy-preserving federated learning," in *Proc. 12th ACM Workshop Artif. Intell. Security*, London, U.K., Nov. 2019, pp. 1–11.

[107] K. Wei *et al.* "Federated learning with differential privacy: Algorithms and performance analysis," *IEEE Trans. Inf. Forensics Security*, vol. 15, pp. 3454–3469, 2020.

[108] A. Girgis, D. Data, S. Diggavi, P. Kairouz, and A. T. Suresh, "Shuffled model of differential privacy in federated learning," in *Proc. Int. Conf. Artif. Intell. Stat.*, 2021, pp. 2521–2529.

[109] N. Phan, X. Wu, H. Hu, and D. Dou, "Adaptive Laplace mechanism: Differential privacy preservation in deep learning," in *Proc. IEEE Int. Conf. Data Min. (ICDM)*, 2017, pp. 385–394.

[110] F. McSherry and K. Talwar, "Mechanism design via differential privacy," in *Proc. 48th Annu. IEEE Symp. Found. Comput. Sci. (FOCS)*, 2007, pp. 94–103.

[111] C. Ma *et al.*, "On safeguarding privacy and security in the framework of federated learning," *IEEE Netw.*, vol. 34, no. 4, pp. 242–248, Jul./Aug. 2020.

[112] C. Dwork and A. Roth, "The algorithmic foundations of differential privacy," *Found. Trends Theor. Comput. Sci.*, vol. 9, nos. 3–4, pp. 211–407, 2014.

[113] Kaspersky. *What Is Cyber Security?* Accessed: Jan. 20, 2021. [Online]. Available: https://www.kaspersky.com/resource-center/definitions/what-is-cyber-security

[114] CISCO. *What Is Cyber Security?* Accessed: Jan. 20, 2021. [Online]. Available: https://www.cisco.com/c/en/us/products/security/what-is-cybersecurity.html

[115] J.-Y. Lee, W.-C. Lin, and Y.-H. Huang, "A lightweight authentication protocol for Internet of Things," in *Proc. Int. Symp. Next-Gener. Electron. (ISNE)*, Mashhad, Iran, 2014, pp. 1–2.

[116] E. Lara, L. Aguilar, M. A. Sanchez, and J. A. García, "Lightweight authentication protocol for M2M communications of resource-constrained devices in industrial Internet of Things," *Sensors*, vol. 20, no. 2, p. 501, Jan. 2020.

[117] S. Arasteh, S. F. Aghili, and H. Mala, "A new lightweight authentication and key agreement protocol for Internet of Things," in *Proc. 13th Int. Iran. Soc. Cryptol. Conf. Inf. Security Cryptol.*, Nov. 2016, pp. 52–59.

[118] A. Haenel, Y. Haddad, and Z. Zhang, "Lightweight authentication scheme for Internet of Things," in *Proc. IEEE Annu. Consum. Commun. Netw. Conf.*, London, U.K., Jun. 2020, pp. 1–2.

[119] J. Han and J. Kim, "A lightweight authentication mechanism between IoT devices," in *Proc. Int. Conf. Inf. Commun. Technol. Convergence*, Jeju, South Korea, Dec. 2017, pp. 1153–1155.

[120] S. Kaushik and C. Gandhi, "Cloud computing security: Attacks, threats, risk and solutions," *Int. J. Netw. Virtual Org.*, vol. 19, no. 1, pp. 50–71, Jul. 2018.

[121] H. Li, F. Li, C. Song, and Y. Yan, "Towards smart card based mutual authentication schemes in cloud computing," *KSII Trans. Internet Inf. Syst.*, vol. 9, no. 7, pp. 2719–2735, Jul. 2015.

[122] Q. Jiang, J. Ma, and F. Wei, "On the security of a privacy-aware authentication scheme for distributed mobile cloud computing services," *IEEE Syst. J.*, vol. 12, no. 2, pp. 2039–2042, Jun. 2018.

[123] L. Barreto, A. Celesti, M. Villari, M. Fazio, and A. Puliafito, "Security and IoT cloud federation: Design of authentication schemes," in *Proc. Int. Internet Things Summit*, Rome, Italy, Oct. 2015, pp. 337–346.

[124] X. Zhang, X. Zhu, J. Wang, H. Yan, H. Chen, and W. Bao, "Federated learning with adaptive communication compression under dynamic bandwidth and unreliable networks," *Inf. Sci.*, vol. 540, pp. 242–262, Nov. 2020.

[125] S. Niknam, H. S. Dhillon, and J. H. Reed, "Federated learning for wireless communications: Motivation, opportunities, and challenges," *IEEE Commun. Mag.*, vol. 58, no. 6, pp. 46–51, Jun. 2020.

[126] A. Nagar, "Privacy-preserving blockchain based federated learning with differential data sharing," 2019. [Online]. Available: arXiv:1912.04859.

[127] X. Qu, S. Wang, Q. Hu, and X. Cheng, "Proof of federated learning: A novel energy-recycling consensus algorithm," 2019. [Online]. Available: arXiv:1912.11745.

[128] S. Jere, Q. Fan, B. Shang, L. Li, and L. Liu, "Federated learning in mobile edge computing: An edge-learning perspective for beyond 5G," 2020. [Online]. Available: arXiv:2007.08030.

[129] Y. Zhan, P. Li, and S. Guo, "Experience-driven computational resource allocation of federated learning by deep reinforcement learning," in *Proc. IPDPS*, New Orleans, LA, USA, Jul. 2020, pp. 18–22.

[130] M. Chen, Z. Yang, W. Saad, C. Yin, H. V. Poor, and S. Cui, "Performance optimization of federated learning over wireless networks," in *Proc. IEEE Glob. Commun. Conf.*, Waikoloa, HI, USA, Dec. 2019, pp. 1–6.

[131] A. Ndikumana *et al.*, "Joint communication, computation, caching, and control in big data multi-access edge computing," *IEEE Trans. Mobile Comput.*, vol. 19, no. 6, pp. 1359–1374, Jun. 2020.

[132] S. R. Pandey, N. H. Tran, M. Bennis, Y. K. Tun, A. Manzoor, and C. S. Hong, "A crowdsourcing framework for on-device federated learning," *IEEE Trans. Wireless Commun.*, vol. 19, no. 5, pp. 3241–3256, May 2020.

[133] H. Kim, J. Park, M. Bennis, and S.-L. Kim, "Blockchained on-device federated learning," *IEEE Commun. Lett.*, vol. 24, no. 6, pp. 1279–1283, Jun. 2020.

[134] M. M. Wadu, S. Samarakoon, and M. Bennis, "Federated learning under channel uncertainty: Joint client scheduling and resource allocation," 2020. [Online]. Available: arXiv:2002.00802.

[135] J. Konečnỳ, H. B. McMahan, D. Ramage, and P. Richtárik, "Federated optimization: Distributed machine learning for on-device intelligence," 2016. [Online]. Available: arXiv:1610.02527.

[136] J. Konečnỳ, Z. Qu, and P. Richtárik, "Semi-stochastic coordinate descent," *Optim. Methods Softw.*, vol. 32, no. 5, pp. 993–1005, Mar. 2017.

[137] F. Ang, L. Chen, N. Zhao, Y. Chen, W. Wang, and F. R. Yu, "Robust federated learning with noisy communication," *IEEE Trans. Commun.*, vol. 68, no. 6, pp. 3452–3464, Jun. 2020.

[138] T. H. T. Le, N. H. Tran, Y. K. Tun, Z. Han, and C. S. Hong, "Auction based incentive design for efficient federated learning in cellular wireless networks," in *Proc. IEEE Wireless Commun. Netw. Conf.*, Seoul, South Korea, Jun. 2020, pp. 1–6.

[139] J. Kang, Z. Xiong, D. Niyato, S. Xie, and J. Zhang, "Incentive mechanism for reliable federated learning: A joint optimization approach to combining reputation and contract theory," *IEEE Internet Things J.*, vol. 6, no. 6, pp. 10700–10714, Dec. 2019.

[140] P. K. Sharma, J. H. Park, and K. Cho, "Blockchain and federated learning-based distributed computing defence framework for sustainable society," *Sustain. Cities Soc.*, vol. 59, Aug. 2020, Art. no. 102220.

[141] G. Lee, J. Park, W. Saad, and M. Bennis, "Performance analysis of blockchain systems with wireless mobile miners," *IEEE Netw. Lett.*, vol. 2, no. 3, pp. 111–115, Sep. 2020.

[142] T. Alladi, V. Chamola, N. Sahu, and M. Guizani, "Applications of blockchain in unmanned aerial vehicles: A review," *Veh. Commun.*, vol. 23, Jun. 2020, Art. no. 100249.

[143] T. Zeng, O. Semiari, M. Mozaffari, M. Chen, W. Saad, and M. Bennis, "Federated learning in the sky: Joint power allocation and scheduling with UAV swarms," 2020. [Online]. Available: arXiv:2002.08196.

[144] R. Fan, J. Cui, S. Jin, K. Yang, and J. An, "Optimal node placement and resource allocation for UAV relaying network," *IEEE Commun. Lett.*, vol. 22, no. 4, pp. 808–811, Apr. 2018.

[145] M. Mozaffari, A. T. Z. Kasgari, W. Saad, M. Bennis, and M. Debbah, "Beyond 5G with UAVs: Foundations of a 3D wireless cellular network," *IEEE Trans. Wireless Commun.*, vol. 18, no. 1, pp. 357–372, Jan. 2019.

[146] M. Mozaffari, W. Saad, M. Bennis, Y.-H. Nam, and M. Debbah, "A tutorial on UAVs for wireless networks: Applications, challenges, and open problems," *IEEE Commun. Surveys Tuts.*, vol. 21, no. 3, pp. 2334–2360, 3rd Quart., 2019.

[147] K. Bonawitz *et al.*, "Towards federated learning at scale: System design," 2019. [Online]. Available: arXiv:1902.01046.

[148] Y. Park, H. Yang, T. Dinh, and Y. Kim, "Design and implementation of a container-based virtual client architecture for interactive digital signage systems," *Int. J. Distrib. Sens. Netw.*, vol. 13, no. 7, pp. 1–14, Jul. 2017.

[149] S. Wu, C. Niu, J. Rao, H. Jin, and X. Dai, "Container-based cloud platform for mobile computation offloading," in *Proc. IEEE Int. Parallel Distrib. Process. Symp.*, Orlando, FL, USA, Jul. 2017, pp. 123–132.

[150] H. Kang, M. Le, and S. Tao, "Container and microservice driven design for cloud infrastructure DeVops," in *Proc. IEEE Int. Conf. Cloud Eng. (IC2E)*, Berlin, Germany, Jun. 2016, pp. 202–211.

[151] Z. Zhong, K. Chen, X. Zhai, and S. Zhou, "Virtual machine-based task scheduling algorithm in a cloud computing environment," *Tsinghua Sci. Technol.*, vol. 21, no. 6, pp. 660–667, Dec. 2016.

[152] H. A. Lagar-Cavilla *et al.*, "SnowFlock: Rapid virtual machine cloning for cloud computing," in *Proc. 4th ACM Eur. Conf. Comput. Syst.*, Nuremberg, Germany, Apr. 2009, pp. 1–12.

[153] V. Smith, C.-K. Chiang, M. Sanjabi, and A. S. Talwalkar, "Federated multi-task learning," in *Proc. Adv. Neural Inf. Process. Syst.*, Long Beach, CA, USA, Dec. 2017, pp. 4424–4434.

[154] X. Liang, Y. Liu, T. Chen, M. Liu, and Q. Yang, "Federated transfer reinforcement learning for autonomous driving," 2019. [Online]. Available: arXiv:1910.06001.

[155] I. Yaqoob, L. U. Khan, S. A. Kazmi, M. Imran, N. Guizani, and C. S. Hong, "Autonomous driving cars in smart cities: Recent advances, requirements, and challenges," *IEEE Netw.*, vol. 34, no. 1, pp. 174–181, Jan./Feb. 2019.

[156] C. Szegedy *et al.*, "Going deeper with convolutions," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, Boston, MA, USA, Jun. 2015, pp. 1–9.

[157] V. Sze, Y.-H. Chen, J. Emer, A. Suleiman, and Z. Zhang, "Hardware for machine learning: Challenges and opportunities," in *Proc. IEEE Custom Integr. Circuits Conf.*, Austin, TX, USA, Mar. 2017, pp. 1–8.

[158] M. Horowitz, "1.1 computing's energy problem (and what we can do about it)," in *IEEE Int. Solid-State Circuits Conf. Dig. Tech. Papers*, Mar. 2014, pp. 10–14.

[159] L. Ceze, M. D. Hill, and T. F. Wenisch, "Arch2030: A vision of computer architecture research over the next 15 years," 2016. [Online]. Available: arXiv:1612.03182.

[160] N. P. Jouppi *et al.*, "In-datacenter performance analysis of a tensor processing unit," in *Proc. 44th Annu. Int. Symp. Comput. Archit.*, Jun. 2017, pp. 1–12.

[161] R. G. Kim, J. R. Doppa, and P. P. Pande, "Machine learning for design space exploration and optimization of manycore systems," in *Proc. IEEE/ACM Int. Conf. Comput.-Aided Design*, San Deigo, CA, USA, Nov. 2018, pp. 1–6.

[162] C. Liu *et al.*, "Progressive neural architecture search," in *Proc. Eur. Conf. Comput. Vis. (ECCV)*, Munich, Germany, Sep. 2018, pp. 19–34.

[163] H. Pham, M. Y. Guan, B. Zoph, Q. V. Le, and J. Dean, "Efficient neural architecture search via parameter sharing," 2018. [Online]. Available: arXiv:1802.03268.

[164] C. Ying, A. Klein, E. Christiansen, E. Real, K. Murphy, and F. Hutter, "NAS-bench-101: Towards reproducible neural architecture search," in *Proc. Int. Conf. Mach. Learn.*, Long Beach, CA, USA, 2019, pp. 7105–7114.

[165] M. A. Rahman. *Neural Architecture Search (NAS)—The Future of Deep Learning*. Accessed: Oct. Jun. 14, 2020. [Online]. Available: https://towardsdatascience.com/neural-architecture-search-nas-the-future-of-deep-learning-c99356351136

[166] T. Elsken, J. H. Metzen, and F. Hutter, "Neural architecture search: A survey," 2018. [Online]. Available: arXiv:1808.05377.

[167] S. Hardy *et al.*, "Private federated learning on vertically partitioned data via entity resolution and additively homomorphic encryption," 2017. [Online]. Available: arXiv:1711.10677.

[168] C. Borrego, M. Amadeo, A. Molinaro, and R. H. Jhaveri, "Privacy-preserving forwarding using homomorphic encryption for information-centric wireless ad hoc networks," *IEEE Commun. Lett.*, vol. 23, no. 10, pp. 1708–1711, Oct. 2019.

[169] X. Li, D. Chen, C. Li, and L. Wang, "Secure data aggregation with fully homomorphic encryption in large-scale wireless sensor networks," *Sensors*, vol. 15, no. 7, pp. 15952–15973, Jul. 2015.

[170] R. Shrestha and S. Kim, "Integration of IoT with blockchain and homomorphic encryption: Challenging issues and opportunities," *Adv. Comput.*, vol. 115, pp. 293–331, Jul. 2019.

[171] T. Lepoint and M. Naehrig, "A comparison of the homomorphic encryption schemes FV and YASHE," in *Proc. Int. Conf. Cryptol. Africa*, Marrakesh, Morocco, May 2014, pp. 318–335.

[172] K. Bonawitz *et al.*, "Practical secure aggregation for privacy-preserving machine learning," in *Proc. ACM SIGSAC Conf. Comput. Commun. Security*, Dec. 2017, pp. 1175–1191.

[173] A. K. Bairagi, M. S. Munir, M. Alsenwi, N. H. Tran, and C. S. Hong, "A matching based coexistence mechanism between eMBB and URLLC in 5G wireless networks," in *Proc. 34th ACM/SIGAPP Symp. Appl. Comput.*, New York, NY, USA, Apr. 2019, pp. 2377–2384.

[174] Q.-V. Pham, T. Leanh, N. H. Tran, B. J. Park, and C. S. Hong, "Decentralized computation offloading and resource allocation for mobile-edge computing: A matching game approach," *IEEE Access*, vol. 6, pp. 75868–75885, 2018.

[175] S. A. Kazmi, N. H. Tran, T. M. Ho, and C. S. Hong, "Hierarchical matching game for service selection and resource purchasing in wireless network virtualization," *IEEE Commun. Lett.*, vol. 22, no. 1, pp. 121–124, Jan. 2018.
[176] S. M. A. Kazmi *et al.*, "Mode selection and resource allocation in device-to-device communications: A matching game approach," *IEEE Trans. Mobile Comput.*, vol. 16, no. 11, pp. 3126–3141, Nov. 2017.
[177] N. Shlezinger, M. Chen, Y. C. Eldar, H. V. Poor, and S. Cui, "UVeQFed: Universal vector quantization for federated learning," 2020. [Online]. Available: arXiv:2006.03262.
[178] A. Reisizadeh, A. Mokhtari, H. Hassani, A. Jadbabaie, and R. Pedarsani, "FedPAQ: A communication-efficient federated learning method with periodic averaging and quantization," in *Proc. Int. Conf. Artif. Intell. Stat.*, Aug. 2020, pp. 2021–2031.
[179] X. Dai, X. Yan, K. Zhou, K. K. Ng, J. Cheng, and Y. Fan, "Hypersphere quantization: Communication-efficient SGD for federated learning," 2019. [Online]. Available: arXiv:1911.04655.
[180] N. Kato, B. Mao, F. Tang, Y. Kawamoto, and J. Liu, "Ten challenges in advancing machine learning technologies toward 6G," *IEEE Wireless Commun.*, vol. 27, no. 3, pp. 96–103, Jun. 2020.
[181] I. Akyildiz, A. Kak, and S. Nie, "6G and beyond: The future of wireless communications systems," *IEEE Access*, vol. 8, pp. 133995–134030, 2020.
[182] M. Wang, T. Zhu, T. Zhang, J. Zhang, S. Yu, and W. Zhou, "Security and privacy in 6G networks: New areas and new challenges," *Digit. Commun. Netw.*, vol. 6, no. 3, pp. 281–291, 2020, doi: 10.1016/j.dcan.2020.07.003.

Latif U. Khan received the M.S. degree (with Distinction) in electrical engineering from the University of Engineering and Technology (UET), Peshawar, Pakistan, in 2017. He is currently pursuing the Ph.D. degree in computer science and engineering with Kyung Hee University (KHU), South Korea. He is working as a Leading Researcher with the Intelligent Networking Laboratory under a project jointly funded by the prestigious Brain Korea 21st Century Plus and Ministry of Science and ICT, South Korea. Prior to joining the KHU, he has served as a Faculty Member and a Research Associate with UET. He has published his works in highly reputable conferences and journals. He has authored/coauthored of two conference best paper awards. He is also author of two books, such as *Network Slicing for 5G and Beyond Networks* and *Federated Learning for Wireless Networks*. His research interests include analytical techniques of optimization and game theory to edge computing, end-to-end network slicing, and federated learning for wireless networks.

Walid Saad (Fellow, IEEE) received the Ph.D. degree from the University of Oslo, Oslo, Norway, in 2010. He is currently a Professor with the Department of Electrical and Computer Engineering, Virginia Tech, Blacksburg, VA, USA, where he leads the Network Science, Wireless, and Security Laboratory. His research interests include wireless networks, machine learning, game theory, security, unmanned aerial vehicles, cyber–physical systems, and network science. He was named the Stephen O. Lane Junior Faculty Fellow with Virginia Tech from 2015 to 2017. In 2017, he was named as the College of Engineering Faculty Fellow. He was a recipient of the NSF CAREER Award in 2013, the Air Force Office of Scientific Research Summer Faculty Fellowship in 2014, and the Young Investigator Award from the Office of Naval Research in 2015. He was the author/coauthor of eight conference best paper awards at WiOpt in 2009, the International Conference on Internet Monitoring and Protection in 2010, the IEEE Wireless Communications and Networking Conference in 2012, the IEEE International Symposium on Personal, Indoor and Mobile Radio Communications in 2015, the IEEE SmartGridComm in 2015, EuCNC in 2017, IEEE GLOBECOM in 2018, and the International Federation for Information Processing International Conference on New Technologies, Mobility and Security in 2019. He was also a recipient of the 2015 Fred W. Ellersick Prize from the IEEE Communications Society, the 2017 IEEE ComSoc Best Young Professional in Academia Award, the 2018 IEEE ComSoc Radio Communications Committee Early Achievement Award, and the 2019 IEEE ComSoc Communication Theory Technical Committee. He received the Dean's Award for Research Excellence from Virginia Tech in 2019. He also serves as an Editor for the IEEE TRANSACTIONS ON WIRELESS COMMUNICATIONS, the IEEE TRANSACTIONS ON MOBILE COMPUTING, and the IEEE TRANSACTIONS ON COGNITIVE COMMUNICATIONS AND NETWORKING. He is also an Editor-at-Large of the IEEE TRANSACTIONS ON COMMUNICATIONS. He is also an IEEE Distinguished Lecturer. (Based on document published on 18 May 2020).

Zhu Han (Fellow, IEEE) received the B.S. degree in electronic engineering from Tsinghua University in 1997, and the M.S. and Ph.D. degrees in electrical and computer engineering from the University of Maryland, College Park, in 1999 and 2003, respectively.

From 2000 to 2002, he was an R&D Engineer of JDSU, Germantown, Maryland. From 2003 to 2006, he was a Research Associate with the University of Maryland. From 2006 to 2008, he was an Assistant Professor with Boise State University, Boise, ID, USA. He is currently a John and Rebecca Moores Professor with the Electrical and Computer Engineering Department and the Computer Science Department, University of Houston, Houston, TX, USA. His research interests include wireless resource allocation and management, wireless communications and networking, game theory, big data analysis, security, and smart grid. He received the NSF Career Award in 2010, the Fred W. Ellersick Prize of the IEEE Communication Society in 2011, the EURASIP Best Paper Award for the *Journal on Advances in Signal Processing* in 2015, the IEEE Leonard G. Abraham Prize in the Field of Communications Systems (Best Paper Award in IEEE JSAC) in 2016, and several best paper awards in IEEE conferences. He is also the winner of 2021 IEEE Kiyo Tomiyasu Award, for outstanding early to mid-career contributions to technologies holding the promise of innovative applications, with the following citation: "for contributions to game theory and distributed management of autonomous communication networks." He has been 1% highly cited researcher since 2017 according to Web of Science. He was an IEEE Communications Society Distinguished Lecturer from 2015 to 2018, and has been an AAAS Fellow since 2019 and an ACM Distinguished Member since 2019.

Ekram Hossain (Fellow, IEEE) is a Professor with the Department of Electrical and Computer Engineering, University of Manitoba, Winnipeg, MB, Canada. He was listed as a Clarivate Analytics Highly Cited Researcher in Computer Science in 2017, 2018, 2019, and 2020. He served as the Editor-in-Chief for the IEEE COMMUNICATIONS SURVEYS & TUTORIALS from 2012 to 2016. He currently serves as the Editor-in-Chief of IEEE Press. He is a member (Class of 2016) of the College of the Royal Society of Canada, and a Fellow of the Canadian Academy of Engineering and the Engineering Institute of Canada. He was elevated to an IEEE Fellow "for contributions to spectrum management and resource allocation in cognitive and cellular radio networks."

Choong Seon Hong (Senior Member, IEEE) received the B.S. and M.S. degrees in electronic engineering from Kyung Hee University, Seoul, South Korea, in 1983 and 1985, respectively, and the Ph.D. degree from Keio University, Tokyo, Japan, in 1997. In 1988, he joined KT, Gyeonggi, South Korea, where he was involved in broadband networks as a member of the Technical Staff. Since 1993, he has been with Keio University. He was with the Telecommunications Network Laboratory, KT, as a Senior Member of Technical Staff, and as the Director of the Networking Research Team until 1999. Since 1999, he has been a Professor with the Department of Computer Science and Engineering, Kyung Hee University. His research interests include future Internet, intelligent edge computing, network management, and network security. He has served as the General Chair, the TPC Chair/Member, or an Organizing Committee Member of international conferences, such as the Network Operations and Management Symposium, the International Symposium on Integrated Network Management, the Asia–Pacific Network Operations and Management Symposium, the End-to-End Monitoring Techniques and Services, the IEEE Consumer Communications and Networking Conference, the Assurance in Distributed Systems and Networks, the International Conference on Parallel Processing, the Data Integration and Mining, the World Conference on Information Security Applications, the Broadband Convergence Network, the Telecommunication Information Networking Architecture, the International Symposium on Applications and the Internet, and the International Conference on Information Networking. He was an Associate Editor of the IEEE TRANSACTIONS ON NETWORK AND SERVICE MANAGEMENT and the IEEE JOURNAL OF COMMUNICATIONS AND NETWORKS. He currently serves as an Associate Editor for the *International Journal of Network Management*. He is a member of the Association for Computing Machinery, the Institute of Electronics, Information and Communication Engineers, the Information Processing Society of Japan, the Korean Institute of Information Scientists and Engineers, the Korean Institute of Communications and Information Sciences, the Korean Information Processing Society, and the Open Standards and ICT Association.