Federated Machine Learning: Survey, Multi-Level Classification, Desirable Criteria and Future Directions in Communication and Networking Systems

Omar Abdel Wahab, Azzam Mourad, Hadi Otrok and Tarik Taleb

Abstract—The communication and networking field is hungry for machine learning decision-making solutions to replace the traditional model-driven approaches that proved to be not rich enough for seizing the ever-growing complexity and heterogeneity of the modern systems in the field. Traditional machine learning solutions assume the existence of (cloud-based) central entities that are in charge of processing the data. Nonetheless, the difficulty of accessing private data, together with the high cost of transmitting raw data to the central entity gave rise to a decentralized machine learning approach called *Federated Learning*. The main idea of federated learning is to perform an on-device collaborative training of a single machine learning model without having to share the raw training data with any third-party entity. Although few survey articles on federated learning already exist in the literature, the motivation of this survey stems from three essential observations. The first one is the lack of a fine-grained multi-level classification of the federated learning literature, where the existing surveys base their classification on only one criterion or aspect. The second observation is that the existing surveys focus only on some common challenges, but disregard other essential aspects such as reliable client selection, resource management and training service pricing. The third observation is the lack of explicit and straightforward directives for researchers to help them design future federated learning solutions that overcome the state-of-the-art research gaps. To address these points, we first provide a comprehensive tutorial on federated learning and its associated concepts, technologies and learning approaches. We then survey and highlight the applications and future directions of federated learning in the domain of communication and networking. Thereafter, we design a three-level classification scheme that first categorizes the federated learning literature based on the high-level challenge that they tackle. Then, we classify each high-level challenge into a set of specific low-level challenges to foster a better understanding of the topic. Finally, we provide, within each low-level challenge, a fine-grained classification based on the technique used to address this particular challenge. For each category of high-level challenges, we provide a set of desirable criteria and future research directions that are aimed to help the research community design innovative and efficient future solutions. To the best of our knowledge, our survey is the most comprehensive in terms of challenges and techniques it covers and the most fine-grained in terms of the multi-level classification scheme it presents.

Index Terms—Federated Learning; Federated Learning Tutorial; Multi-Level Classification; Statistical Challenges; Transfer Learning; Machine Learning; Security; Communication and Networking Systems.

1 Introduction

The fast-growing adoption of Internet of Things (IoT) and social networking applications is leading to an unprecedented growth in the volumes of data that are generated on a daily basis. In particular, the International Data Corporation (IDC) anticipates that, by 2025, there will be 79ZB of data created by billions of IoT devices, pushing organizations to rethink their data governance, retention, and usage strategies. Storing and analyzing such large volumes of data has long been done on the cloud, owing to the large number of advantages that the cloud computing technology provides, such as cost efficiency and unlimited computing and storage capabilities [1], [2], [3]. Nonetheless, due to the ever-rising data privacy concerns and network limitations, a pure centralized cloud-based data storage and analytics approach becomes unrealistic. In fact, data owners often feel concerned about sharing their data with a third-party whether it is a well-known organization or mysterious to them. In this context, strict legislations such as the US Consumer Privacy Bill of Rights[1] and the European Commission's General Data Protection Regulation (GDPR)[2] have been designed to protect users' privacy. For instance, the Articles 5 and 6 of the GDPR restrict the data collection and storage to only what is user-consented and decidedly indispensable for processing. Moving to the network limitation problem and emergency of low-latency applications requiring fast analysis, the fact that the cloud data centers are often deployed in locations that are far from those of the data owners leads to high data processing delays due to the long-distance communications. In the light of these two crucial factors, the trend in data storage and analysis is shifting from being cloud-based and centralized to being distributed and on-device [4], [5]. The key enabler technology for such a shift is that of *edge computing* [6], [7], wherein edge nodes such as smartphones, sensor, micro servers, autonomous vehicles and home gateways are supplied with computing and storage capabilities to enable them to host and analyze data locally within minimal delay. Edge nodes then periodically communicate with the cloud servers to send them the processed data for historical and long-term storage.

O. Abdel Wahab is with the Department of Computer Science and Engineering, Université du Québec en Outaouais, Gatineau, Canada (e-mail: omar.abdulwahab@uqo.ca).
H. Otrok is with the Department of EECS, Center on Cyber-Physical Systems, Khalifa University of Science and Technology, Abu Dhabi, UAE (e-mail: Hadi.Otrok@ku.ac.ae).
A. Mourad is with the Department of Mathematics and Computer Science, Lebanese American University, Beirut, Lebanon (e-mail: azzam.mourad@lau.edu.lb).
T. Taleb is with the Department of Communications and Networking, Aalto University, Espoo 02150, Finland, with the Centre for Wireless Communications (CWC), University of Oulu, Oulu 90570, Finland, and with the Department of Computer and Information Security, Sejong University, Seoul 05006, South Korea (e-mail: tarik.taleb@aalto.fi).

1. https://www.congress.gov/bill/116th-congress/senate-bill/2968/text
2. https://gdpr.eu/data-privacy/

In order to make this idea feasible, it was necessary to adapt the machine learning process to this vision in order to enable what is known as the *machine learning at the edge*. In this context, the new paradigm of *Federated Learning (FL)* has been proposed in 2016 by McMahan et al. [8] to enable local and distributed machine learning training at the level of edge nodes or end devices. The main idea of federated learning is to enable a large number of edge devices or servers storing local data observations, called *clients*, to locally and collaboratively train one single machine learning model without having to share their raw data. A coordinating server (often called *parameter server*[3]) then aggregates the contributions from all the clients, derives an updated model and shares this model with the participating clients to benefit from their learning experience and to enable them to pursue their local training in future iterations. Federated learning substantially differs from the centralized (cloud-based) machine learning paradigm and poses additional unique challenges in the following aspects [9]:

Privacy: In federated learning, the raw data never leaves the user's device since the training is done locally on each device. Nonetheless, having more users involved in one collaborative model increases the risk of launching inference attacks that aim to infer sensitive information from the users' training data;
Communication: In federated learning, no raw data need to be communicated with any central server, which reduces the amount of information that needs to be transmitted over the network. However, since the machine learning model is trained collaboratively, many model updates need to be communicated between the clients and the server over many iterations, which poses additional communication costs.
Latency: With federated learning, the decision-making models are trained locally on the edge/end devices instead of being sent to the cloud, leading to lower latency and waiting times.
Statistical Heterogeneity: Given that the training data on each client device depends on its own usage patterns, the local dataset of one client in federated learning is not expected to be representative of the overall data distribution. Similarly, as clients use their services or applications in varying degrees, the local datasets across clients tend to have varying sizes.
Massive Distribution: The number of clients that participate in the federated training is expected to be significantly larger than the average number of training samples per client.
Connectivity: In federated learning, client devices are frequently offline or on slow or expensive connection. This means that the connectivity in federated learning is limited and that the process of selecting clients to participate in the federated training might be biased toward certain conditions (e.g., local time zone, device being charged or not, etc).

From the technical perspective, federated learning can be implemented using two main strategies: Horizontal Federated Learning (HFL) and Vertical Federated Learning (VFL) [10]. In HFL, the participating client devices share the same set of features but target different populations. An example of HFL could be two banks operating in the same country. Even though the clientele of the banks is non-overlapping, their data are likely to have a similar feature space since they adopt similar business models and operate in the same country. In VFL, the client devices share the same population but target different sets of features. An example of VFL is two companies offering two different services (e.g., counselling and shipping) but having a large intersection at the level of the clienteles. Such companies might be interested in cooperating on the (distinct) feature spaces they own to gain each a better understanding about its own business situation.

3. In the rest of the paper, we use the term *parameter server* to refer to the cental coordinating server.

1.1 Related Work

Few recent survey articles on federated learning have been proposed. We discuss hereafter these surveys and highlight the unique contributions of our work. In [11], the authors provide a detailed survey on the challenges and research directions of federated learning. In particular, they discuss the challenges related to communication efficiency, data privacy, data heterogeneity and model aggregation. In [12], the authors classify the federated learning approaches based on six aspects, i.e., machine learning model, data distribution, communication architecture, privacy mechanism, scale of federation and motivation of federation. In [13], the authors discuss the unique features and challenges of federated learning, offer a broad overview of the literature and highlight several future research directions. In particular, they consider four challenges of federated learning, i.e., communication-efficiency, systems heterogeneity, statistical heterogeneity and privacy. In [10], the authors discuss the definitions, architectures and applications of federated learning framework. They classify the literature of federated learning based on the learning architecture, resulting in three categories: vertical federated learning, horizontal federated learning and federated transfer learning. Different from these surveys, we consider in this work a wider set of challenges such as client selection and scheduling, and service pricing. Moreover, different from these surveys, we provide in this work a more fine-grained three-level classification of the current literature based on the challenge that they address, the sub-challenges that exist within each challenge and the techniques used to address each particular sub-challenge. Furthermore, we define a set of desirable criteria and future research directions that we believe are necessary to address each underlying challenge.

The potential of federated learning in the domains of wireless communication and mobile edge network has been studied in [14] and [15] respectively. The authors of [14] investigate the role of federated learning in the emerging 5G technology. Several use cases that demonstrate how federated learning could be effective in addressing key challenges related to 5G are discussed. In the context of edge computing and content caching, discussions supported with simulation results show that federated learning is an effective means to predicting popular content on mobile devices while preserving the privacy of the users' data. Moving to spectrum management, federated learning can be capitalized on to allow each radio to transfer its local utilization model to a central aggregator, which then leverages these data to create a global learning model. This global model can then be used to derive efficient spectrum access decision-making models. Finally, in the context of 5G core network, vertical federated learning, in which distributed datasets share the same sample space but differ in the feature space, can be used to design intelligent network management techniques. The idea is to allow each entity to manage some specific features (e.g., access mobility management function, session management function, etc.) of the whole dataset that englobes the overall users of the network. Different from our work which addresses the different aspects of federated learning, this survey is restricted to discussing the role of federated learning in the domain of wireless communications. In [15], the authors present a survey that combines the concepts federated learning and Mobile Edge Computing (MEC). After presenting a tutorial on federated learning and explaining its role as an enabling technology for MEC optimization, the authors classify the federated learning approaches into two categories, i.e., federated learning at mobile edge networks and federated learning for mobile edge networks. The first category gathers the approaches that address the challenges of implementing federated training on the edge devices, while the second category gathers the approaches that investigate federated learning as a means for optimizing MECs. These two surveys are restricted to only discussing the potential of federated learning in different aspects of networking, but they

provide no classification of the existing federated learning literature nor desirable criteria for future solutions. In this survey, we believe that, besides illustrating the potential of federated learning in the communication and networking domain, providing a multi-level fine-grained classification of the federated learning literature in general would help researchers in the domain better understand the field which would enable them to design more detailed and efficient solutions.

In [16], the authors survey the current progress on federated learning in the domain of healthcare informatics. They classify the current approaches in terms of statistical challenges, communication efficiency, privacy and security issues. Different from this survey which is specific to the healthcare informatics domain, our survey is oriented to the communication and networking research community. In [17], the authors survey the topic of distributed machine learning with federated learning as an example. The distributed machine learning is divided into three main processes, i.e., machine learning optimizers, distributed optimization and data aggregation. Thereafter, the federated learning framework is introduced and discussed only from the perspective of communication efficiency. Different from this survey, our survey is specific to federated learning, where we address the different aspects and application domains of this emerging concept.

We summarize in Table 1 the main similarities and differences between our survey and the existing surveys on federated learning.

1.2 Contributions

The motivation for this survey stems from four main observations. The first one is that **the existing survey papers focus only on some common challenges of federated learning such as statistical challenges, communication efficiency, security and privacy**. Nonetheless, there exists some other substantial challenges that need further investigation such as service pricing and client selection and scheduling. In this work, we provide a comprehensive survey that considers all these aspects to provide the reader with a holistic view of the federated learning paradigm.

The second observation is the **lack of a fine-grained multi-level classification of the federated learning literature**, where the classification schemes in the existing surveys are based on only one aspect such as the addressed challenges, learning architecture or role of federated learning in a particular application domain (i.e., healthcare and networking). In this work, we take one step ahead and propose a three-level fine-grained classification scheme. First, we classify the federated learning approaches based on the challenge that they address. Then, we classify each corresponding challenge into several specific sub-challenges to enable a better understanding of the topic. Finally, we provide a classification within each sub-challenge based on the technique used to address the underlying sub-challenge. Even though a couple of survey papers [15], [14] discuss the potential of federated learning in the networking domain, these surveys do not provide any classification of the federated learning literature. Different from these papers, our vision in this work is that providing a detailed and fine-grained classification of the broad federated learning literature in an accessible fashion would help the communication and networking research community better understand the tiniest details in the domain. This would enable them to design more thoughtful and to the point solutions. For example, by learning the statistical and security challenges that encounter federated learning along with the techniques that are used in the literature to address them, a researcher in the domain of communication and networking would be able to design a more holistic federated learning-based communication solution that also deals with the non-Independent and Identically Distributed nature of the data and the malicious attacks that can be launched against the distributed training process.

The third observation is the **lack of explicit and elaborate directives for researchers to help them design future federated learning solutions**. We define in this work, for each underlying challenge, a set of desirable criteria and future research directions that we believe are helpful for the success and effectiveness of the future federated learning solutions. In summary, the proposed classification scheme and criteria aim to help (1) readers to easily visualize the current challenges of federated learning along with the state-of-the-art techniques that are employed to address them; (2) research community to have a clear roadmap on how to design prospective solutions based on a set of explicit and well-defined criteria; and (3) beginners in the field to easily grasp the main concepts of federated learning and to be on the lookout for the current trends in this emerging field.

We also provide an accessible tutorial on FL, its alternative learning paradigms (i.e., distributed learning, parallel learning, ensemble learning and gossip learning), and its enabling technologies (i.e., Internet of Things (IoT), cloud computing, edge computing and 5G/6G networks). Additionally, we discuss the applications of federated learning in the domain of communication and networking and highlight some future promising applications of federated learning in this domain.

1.3 Survey Methodology

The approaches chosen to be included in our survey are selected from papers published between 2016 (the year when the concept of federated learning was first introduced) to 2020 in refereed journals and conferences as well as in preprints, resulting in 130 surveyed papers. We believe that we have covered most of the papers that addressed problems related to federated learning. The strategy followed to gather these papers consisted in (1) searching for the keyword "federated learning" on many existing search engines; and (2) tracking the citations of the collected papers to make sure that we cover the articles that may not be returned in the search engine's result set.

The classification scheme consists of three interdependent levels. In the first level, the current federated learning approaches are categorized based on the high-level challenge they address. In the second level, each high-level challenge is broken down into several specific low-level sub-challenges. In the third level, a classification within each sub-challenge is provided based on the technique that is used to deal with that sub-challenge. Note that in some cases, it is possible for an article to appear in more than one category of high-level challenges. For example, if a certain article mainly addresses a statistical challenge of federated learning but also provides a privacy-preservation component, the article would appear under both the *statistical challenges* category and *privacy concerns* category. In such a case, only the statistical part of the article is classified and discussed under the *statistical challenges* category, whereas the privacy part is classified and discussed under the *privacy concerns* category.

The criteria that are defined in this survey have been inspired by our readings of the surveyed papers. We do not claim that our proposed criteria cover all the necessary aspects for improvement; but we believe that these criteria could be quite useful for designing innovative solutions and overcoming some persisting challenges. It is worth mentioning that, in some cases, not all the criteria defined for a particular aspect (e.g., communication efficiency, client selection and scheduling, etc.) need to be met to design an "ideal solution". A subset or a combination of these criteria might be enough to design a good solution.

1.4 Survey Insights

As mentioned earlier, our classification scheme consists of three levels. The first classification level, which is based on the high-level addressed challenge, resulted in six categories, i.e., statistical challenges, communication efficiency, client selection and scheduling, security concerns, privacy concerns and service pricing. This classification scheme is depicted in Fig. 1. We provide in Fig. 2

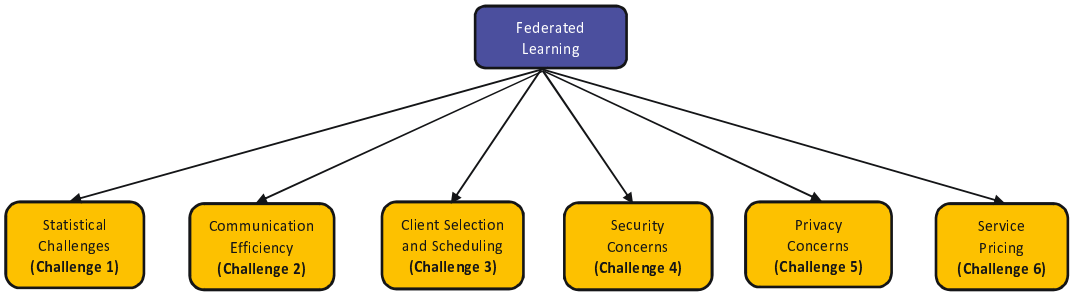


Fig. 1: Classification of the federated learning papers based on the high-level challenge that they address

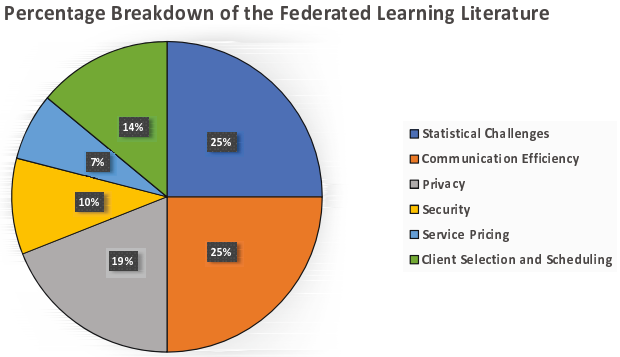


Fig. 2: Percentage breakdown of the federated learning literature: The approaches that address statistical and communication efficiency challenges account for 50% of the existing literature.

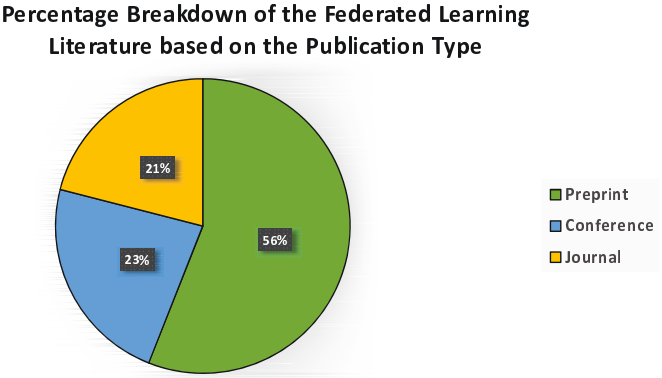


Fig. 3: Percentage breakdown of the federated learning literature based on the publication type: Most of the federated learning papers are published as preprints.

percentage breakdown of the federated learning literature based on the first-level classification scheme. By observing this figure, we conclude that the statistical and communication efficiency challenges have been investigated the most, where each of them accounts for 25% of the surveyed papers. Thereafter comes the privacy concerns category with 19% of the surveyed papers, followed by the client selection and scheduling with 14% of the surveyed papers, the security concerns category with 10% of the surveyed papers, and finally the service pricing with 7% of the surveyed papers. The second and third-level classifications are explained later in the paper in a fine-grained way within each high-level challenge category. We provide in Fig. 3 percentage breakdown of the federated learning literature based on the publication type. We notice from this figure that most federated learning articles (i.e., 56%) are published as preprints. On the other hand, conferences account for 23% of the papers while journals account for 21%. These percentages can be attributed to the fact that federated learning is a relatively new research topic combined with the fact that the publication time can sometimes be long. Therefore, we expect that, in the few coming months and years, many of the surveyed papers will move from preprints to be published as journals or conferences.

1.5 Survey Outline

In Section 2, we explain the concept of federated learning, discuss the commonalities and differences between federated learning and other related learning concepts and finally clarify the links between federated learning and the emerging technologies such as Internet of Things (IoT), cloud computing, edge computing and 5G/6G networks. We also review the applications of federated learning in the communication and networking domain.

TABLE 1: Comparative summary between our survey and the existing surveys on federated learning

Approach	Similarities	Differences
Kairouz et al. [11]	Similar to our survey, this survey addresses some common challenges of federated learning such as Non-IID Data, communication efficiency, user data privacy, security concerns and bias mitigation.	The classification scheme proposed in this survey is quite high-level, where for example the statistical and communication concerns are discussed under one category called "Improving Efficiency and Effectiveness". No classification is provided for the techniques that are proposed to address the discussed challenges. Our work addresses additional challenges that are not mentioned in [11] such as reliable client selection, resource management and training service pricing.
Li et al. [12]	Similar to our survey, this survey addresses some common challenges of federated learning such as privacy concerns and communication efficiency.	This work provides a system-level classification scheme of federated learning that consists of six aspects, which are: data distribution, machine learning model, privacy mechanism, communication architecture, scale of federation, and motivation of federation. The presented classification scheme is not based on a clear criterion, which might be confusing for the reader. For example, communication and privacy are challenges of federated learning, while the scale and motivation of the federation are rather system design choices. Our classification scheme is more systematic, multi-level and based on clear criteria at each level.
Li et al. [13]	Similar to our survey, the classification scheme provided in this survey is multi-level and considers the high-level challenges of federated as well as the sub-challenges within each underlying challenge.	No classification is provided for the techniques that are used to address each specific sub-challenge. Our work addresses challenges of federated learning that are not discussed in [13] such as reliable client selection, resource management and training service pricing.
Yang et al. [10]	Similar to our survey, this survey discusses the privacy concerns of federated learning and explains the concept of federated transfer learning.	The classification scheme presented in this survey is based on the distribution characteristics of the data, resulting in three categories: vertical federated learning, horizontal federated learning and federated transfer learning. Our classification scheme is based on different criteria such as the high-level challenges of federated learning, specific sub-challenges under each high-level challenge and techniques proposed to address each particular sub-challenge.
Niknam et al. [14]	Similar to our survey, this survey is mainly dedicated to the communication and networking community.	This survey focuses on the possible applications of federated learning in the 5G networks and explains the key technical challenges for future research on federated learning in the context of wireless communications. In our survey, we adopt a different approach for presenting the concept of federated learning to the communication and networking community. In particular, besides illustrating the potential of federated learning in the communication and networking domain, we provide a multi-level fine-grained classification of the federated learning literature in general.
Lim et al. [15]	Similar to our survey, this survey is mainly dedicated to the communication and networking community. Similar to our survey as well, this survey takes into consideration some overlooked challenges in other surveys such as client participation motivation and resource allocation.	No sub-classification is provided for the studied challenges based on modular specific sub-challenges. In our work, we uncover several research directions that are not discussed in [15], including but not limited to: Multi-hop routing, utility-based federated training, optimization of interdependent server-client strategies, zero trust security, open market-based federated learning.
Xu et al. [16]	We address some common challenges of federated learning such as statistical challenges, communication efficiency, privacy and security concerns.	This survey is mainly dedicated to the healthcare informatics community, while our survey is dedicated to the communication and networking community. No sub-classification is provided for the studied challenges based on modular specific sub-challenges.
Gu et al. [17]	Similar to our survey, this survey discusses the communication challenges of federated learning and explains the role of edge computing in boosting the applicability of federated learning.	This survey discusses distributed machine learning in general and takes federated learning as only one example. The classification scheme provided is based on the execution processes of distributed machine learning, resulting in three principal classes, i.e., machine learning optimizers, distributed optimization and data aggregation.

Sections 3 to 7 are dedicated to discussing each single high-level challenge depicted in Fig. 1. More specifically, in Section 3, we classify and discuss the statistical challenges of federated learning and highlight some desirable criteria for future solutions. In Section 4, the communication efficiency challenges are classified and discussed and some desirable criteria for future solutions are highlighted. The client selection and scheduling challenges are classified and discussed in Section 5 and desirable criteria for future solutions are highlighted. The security challenges of federated learning are classified and discussed in Section 6 and some desirable criteria for future solutions are highlighted. The privacy challenges of federated learning are classified and discussed in Section 7 and desirable criteria for future solutions are highlighted. In Section 8, we classify and discuss the service pricing challenges of federated learning and highlight some desirable criteria for future solutions. In Section 9, we discuss the future directions in federated learning based on each category of high-level challenges described in our classification scheme (Fig. 1). Finally, in Section 10, we conclude the paper and recapitulate the main insights of the survey.

2 Federated Learning: Definitions, Preliminaries, and Related Concepts

In this section, we first explain the concept of federated learning and that of federated averaging. Thereafter, we explain the main commonalities and differences between federated learning and the

other related learning concepts such as distributed learning, ensemble learning, parallel learning, gossip learning and shared machine learning. Finally, we explain the links between federated learning and several emerging technologies such as Internet of Things (IoT), cloud computing, edge computing and 5G/6G networks.

2.1 Federated Learning

Federated learning is a distributed machine learning approach in which the training of a certain machine learning model is collaboratively done by a number of participants called *clients* over multiple iterations. The concept of federated learning was first introduced in [8] as a distributed training model that is executed by a set of mobile devices that share local model updates with a central server whose role is to aggregate these updates to build a global machine learning model. An aggregation model called *Federated Averaging* (explained in Section 2.1.1) is also introduced to allow the server to combine local stochastic gradients from the different devices using iterative model averaging. A federated learning scenario consists of one central server called *parameter server* and a set of N clients, each having its own local dataset. At the beginning of each federated training iteration, a subset C N of clients is chosen to receive the current global state of the shared model in terms of model weights[4]. Upon receiving the global state, each client uses its own CPU and energy resources to carry out local computations on its own dataset based on the shared parameters. Clients then send the model updates (i.e., the weights that are learned locally based on the client's local dataset) to the parameter server which applies these updates to its current global model to generate a new one. This process is repeated over several iterations (sometimes referred to as *epochs*[5]) until the global model reaches a certain accuracy level determined by the parameter server. In summary, a federated learning scenario consists of two main phases, i.e., *local update* and *global aggregation*. The local update phase refers to the process of computing the gradient descents by the client devices to minimize the underlying loss function with respect to their local data. Global aggregation includes the steps of collecting the updated model parameters by the server from the different client devices, aggregating these parameters and then sending back the aggregate parameters to the clients to be used in their next training iteration. The general idea of federated learning is depicted in Fig. 4.

Formally, assume that a subset C N of clients is selected by the parameter server to participate in the federated training process. Each client c C holds a training dataset D_c X_c Y_c , where X_c ^{D_c} ^{d} represents the feature space vector of c's training data and Y_c ^{D_c} ^{m} is the associated label matrix. To determine the optimal set of parameters that fits the training data, the training model has to optimize a loss (objective) function, which penalizes the model when it produces an inaccurate label on a data point. Let l $\mathbf{W};x_i$ y_i be the loss function for each data sample x_i, with $\mathbf{W}$ being a matrix (or several matrices) of weights between neurons [18]. Correspondingly, the loss function on a client's local dataset D_c is given in Equation (1).

$$f_c \ \mathbf{W} \quad \frac{1}{D_c} \ _{i \ D_c} \ l \ \mathbf{W};x_i \ y_i \tag{1}$$

Let S $^{C}_{c \ 1}$ D_c denote the total number of samples over the C clients. The global loss function over all the C clients and S samples is given in Equation (2).

$$F \ \mathbf{W} \quad \ ^{C}_{c \ 1} \ \frac{D_c}{S} f_c \ \mathbf{W} \tag{2}$$

4. In the following, we refer to the model weights as *model paramaters*.

5. In the rest of the paper, we use the terms *iteration* and *epoch* interchangeably.

The type of the loss function F $\mathbf{W}$ is dependent on the underlying machine learning model [19]. For example, it can be convex in case of a logistic regression model or non-convex in the case of a neural network model.

2.1.1 Federated Averaging

Federated Averaging, also referred to as *FedAvg*, is a global model aggregation algorithm that is executed by the parameter server once it receives the local updates from the clients. Algorithm 1 describes the logic of *FedAvg*. In the first step of the algorithm, the server initializes the parameters $_0$ of the training model (line 3) and solicits the appropriate set of clients to participate in the training (line 5). Thereafter, each client carries out the local training through minimizing the local loss function (Equation (1)) using the SGD algorithm on the mini-batches selected by the algorithm from the training dataset (line 7). Once the model updates from the different clients are received, the server minimizes the global loss function (Equation (2)) through averaging over the received gradients. The federated learning training continues until the global loss function converges or until a desirable accuracy level is attained.

The complexity of the *FedAvg* algorithm is analyzed in [20], where the authors theoretically prove that *FedAvg* converges to the global optimum at a rate of O 1 T for strongly convex and smooth problems, with T being the number of stochastic gradient descent iterations. Another observation is that the impact of E epochs of local updates with a fairly small learning rate is similar to that of one epoch with a larger learning rate. Moreover, the authors demonstrate that the impact of partial client participation (i.e., not all selected clients participating in the training process) is manageable, where it makes the averaged sequence to have a larger variance. The authors argue that this problem can be regulated through adjusting the learning rates of the clients.

Algorithm 1: Federated Averaging Algorithm

1: **Input**: Number of global iterations T
2: **Input**: Number of local training epoches E
3: **Input**: Desired level A_T of model accuracy
4: **Output**: Final parameters $_t$
1: **procedure** FEDAVG
2: **ServerGlobalUpdate:**
3: Initialize parameters $_0$
4: **Repeat**
5: C Select a subset of clients
6: **for each** client c C **do**
7: $^{c}_{t}$ **CientLocalUpdate**($_{t \ 1}$)
8: **end for**
9: $_t$ $C \frac{n^c}{n} \ ^{c}_{t}$
10: **Until** A_T is attained
11: **ClientLocalUpdate**( )
12: **for** local iteration e 1 to E **do**
13: **for each** each batch b in client's split **do**
14: $L \ b$;
15: **end for**
16: **end for**
17: return local model
18: **end procedure**

2.1.2 Variants of Federated Averaging

Several variants of *FedAvg* have been proposed to improve the aggregation of model updates in federated learning. In what follows, we discuss these variants and shed light on their main contributions compared with *FedAvg*.

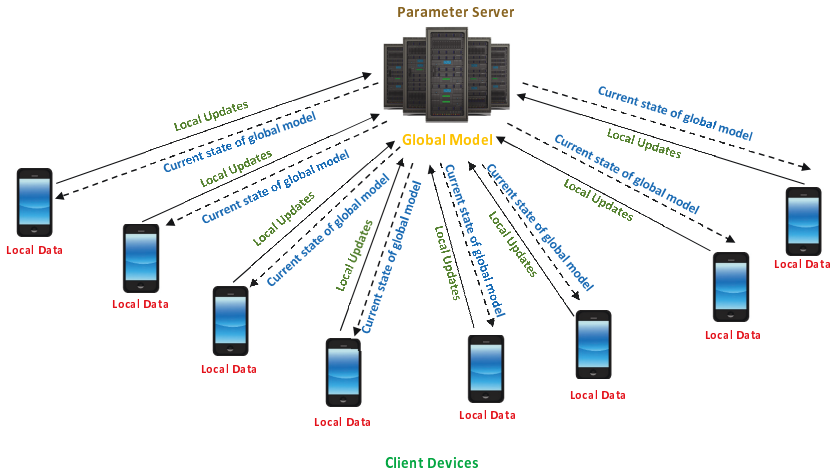


Fig. 4: Federated Learning: The parameter server shares a global machine learning model with the client devices, which use the parameters of the global model to locally train a model on their local data and then share the model updates with the server.

- **FedProx**: *FedProx* [21] is a variation of *FedAvg* that includes a component to represent the statistical heterogeneity across client devices. The main idea of *Fedprox* is to add a proximal term to the local training subproblem on each client device. This term compels the local updates to be closer to the initial global model with the aim of limiting the influence of each local model update on the global model. In all, *FedProx* is proposed to improve the convergence on statistically heterogeneous data. Similar to *FedAvg*, in *FedProx*, all the devices are weighted equally in the global aggregation phase, such that the differences in the device capabilities (e.g., hardware) are not taken into consideration.
- **FedPAQ**: *FedPAQ* [22] is a communication-oriented aggregation technique that suggests to do periodic averaging of the local model updates. Specifically, instead of synchronizing their model updates with the server at each iteration, *FedPAQ* enables clients to carry out multiple local updates on the model prior to sharing the updates with the server. To further reduce the communication overhead, *FedPAQ* allows as well for a partial device participation, where only a subgroup of client devices is chosen to participate in the training at each iteration on the basis of several factors such as the device being (1) connected to a free wireless network, (2) idle and (3) reachable to a base station. Similar to *FedAvg*, the new global model in *FedPAQ* is computed as the average of local models, which entails high complexity in both strongly convex and non-convex settings.
- **Turbo-Aggregate**: *Turbo-Aggregate* [23] is a communication and security-oriented aggregation technique that is based on a multi-group circular strategy for aggregating the model updates. Specifically, clients are divided into multiple groups and at each iteration the clients belonging to one group transmit the aggregated model updates of all clients in the previous groups and the local model updates of the current group to the clients of the next group. *Turbo-Aggregate* includes as well a security component that employs an additive secret sharing mechanism whose main idea is to add some randomness into each local model to preserve the privacy clients' data. The randomness is designed to disappear upon the aggregation of the local models. *Turbo-Aggregate* is quite suitable for wireless topologies, in which network conditions and user availability can vary quickly. Nonetheless, the secure aggregation mechanism embedded in *Turbo-Aggregate*, although effective in handling user dropouts, cannot adapt to new users that join the network. Therefore, it would be interesting to extend it through developing a self-configurable protocol that can accommodate new users on-the-go, by re-configuring the system specifications (i.e., the multi-group structure and coding setup) to ensure that the resilience and privacy guarantees are satisfied.
- **FedMA**: *FedMA* [24] is a statistics-oriented aggregation technique that takes into account the permutation invariance of the neurons in the neural network model before performing the aggregation. The objective is to enable global model size adaptation. *FedMA* employs a Bayesian non-parametric mechanism which allows it adjust the size of the central model to the heterogeneity of data distribution. However, *FedMA* can be vulnerable to model poisoning attack, where an adversary can easily trick the system to expand the global model in order to accommodate any poisoned local model.
- **HierFAVG**: *HierFAVG* [25] is a communication-oriented aggregation technique that aims to foster partial model aggregation at the level of edge servers. Specifically, *HierFAVG* is based on a hierarchical client-edge-cloud architecture whereby each edge server is allowed to aggregate the model updates of its own clients. Subsequently, after a fixed number of model aggregations, the edge-level aggregate models are forwarded to a cloud server for a global aggregation. Such a multi-level structure enables a more efficient model exchange over the existing edge-cloud architecture. However, *HierFAVG* is still vulnerable to the problems of stragglers and end device dropouts.

We summarize in Table 2 the main aggregation techniques in federated learning and classify them based on the federated learning challenge that they try to solve (Fig. 1).

2.2 Federated Learning and Other Learning Approaches

We explain in this section the main points of similarities and points of differences between federated learning and the existing learning approaches, i.e., distributed learning, ensemble learning and parallel learning. We also summarize in Table 3 the main idea of each of these learning approaches.

2.2.1 Distributed Learning

Distributed machine learning or sometimes called *parameter server framework* is a multi-node machine learning approach that is proposed

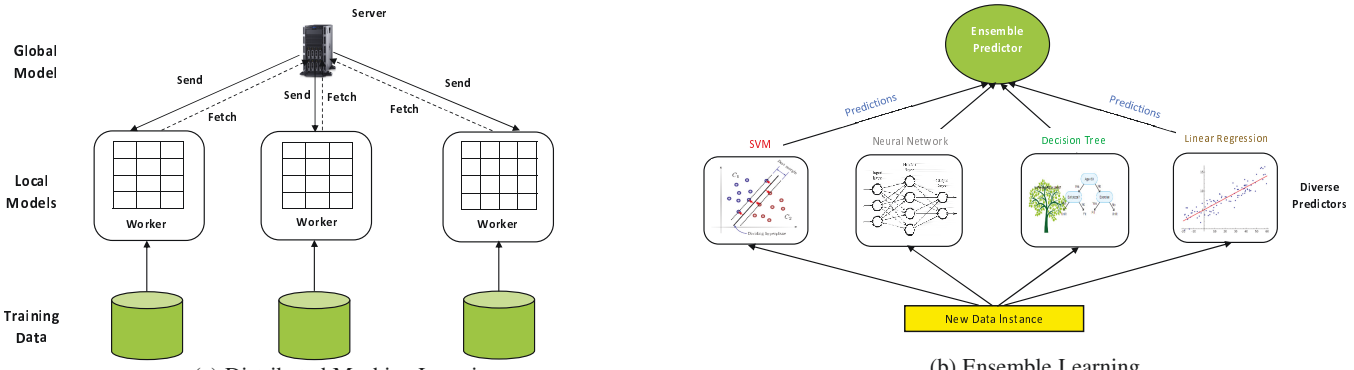
TABLE 2: Comparative summary of the main federated learning aggregation techniques

Aggregation Technique	Classification	Main Idea	Discussion
FedAvg [9]	Statistical	Clients perform several batch updates on their local data and to transmit updated weights rather than the gradients with the server.	From a statistical perspective, *FedAvg* has been shown to start diverging in settings where the data is non-identically distributed across devices. From a system's perspective, *FedAvg* does not allow participating devices to perform variable amounts of local work based on their underlying systems constraints.
FedProx [21]	Statistical	Add a proximal term to the local training subproblem on each client device to limit the influence of each local model update on the global model.	*FedProx* is proposed to improve the convergence on statistically heterogeneous data. Similar to *FedAvg*, in *FedProx*, all devices are weighted equally in the global aggregation phase, in that the differences in the device capabilities (e.g., hardware) are not taken into consideration.
FedPAQ [22]	Communication	Enable clients to carry out multiple local updates on the model prior to sharing the updates with the server.	Similar to *FedAvg*, the new global model in *FedPAQ* is computed as the average of local models, which entails high complexity in both strongly convex and non-convex settings.
Turbo-Aggregate [23]	Communication and security	A multi-group strategy in which the clients are divided into several groups and the model updates are shared among groups in a circular manner. An additive secret sharing mechanism to preserve the privacy clients' data.	Turbo-Aggregate is quite suitable for wireless topologies, in which network conditions and user availability can vary quickly. The secure aggregation mechanism embedded in *Turbo-Aggregate*, although effective in handling user dropouts, cannot adapt to new users that join the network. Therefore, it would be interesting to extend it through developing a self-configurable protocol that can accommodate new users on-the-go, by re-configuring the system specifications (i.e., the multi-group structure and coding setup) to ensure that the resilience and privacy guarantees are satisfied.
FedMA [24]	Statistical	Account for the permutation invariance of the neurons before performing the aggregation to enable global model size adaptation.	Can adjust the size of the central model to the heterogeneity of data distribution using a Bayesian non-parametric mechanism. The Bayesian non-parametric mechanism in FedMA can be vulnerable to model poisoning attack, where an adversary can easily trick the system to expand the global model in order to accommodate any poisoned local model.
HierFAVG [25]	Communication	A hierarchical client-edge-cloud aggregation architecture whereby edge servers aggregate the model updates of their clients and then send them to the cloud server for global aggregation.	This multi-level structure enables a more efficient model exchange over the existing edge-cloud architecture. *HierFAVG* is still vulnerable to the problems of *stragglers* and end device drop-outs.

to improve the performance and scale to larger input data sizes. The main difference between distributed learning and federated learning is that is in the former approach, local workers serve as local data collectors and are not supposed to receive any global model from the parameter server. Instead, these workers train the machine learning model locally and send the parameters back to the server, without waiting to receive any updated global model thereafter. The parameter server organizes the locally trained models to provide a global estimation of the parameters under investigation. From an architectural point of view, federated learning intersects with the concept of *data parallelism* rather than that of *model parallelism* [26] in that each client device executes the same machine learning task on different chunks of the distributed data. On the other hand, in *model parallelism*, each node is supposed to perform a different task on the same dataset. For example, in a deep learning scenario, *model parallelism* could be achieved through splitting the layers of the deep network across devices to be evaluated in parallel.

2.2.2 Ensemble Learning

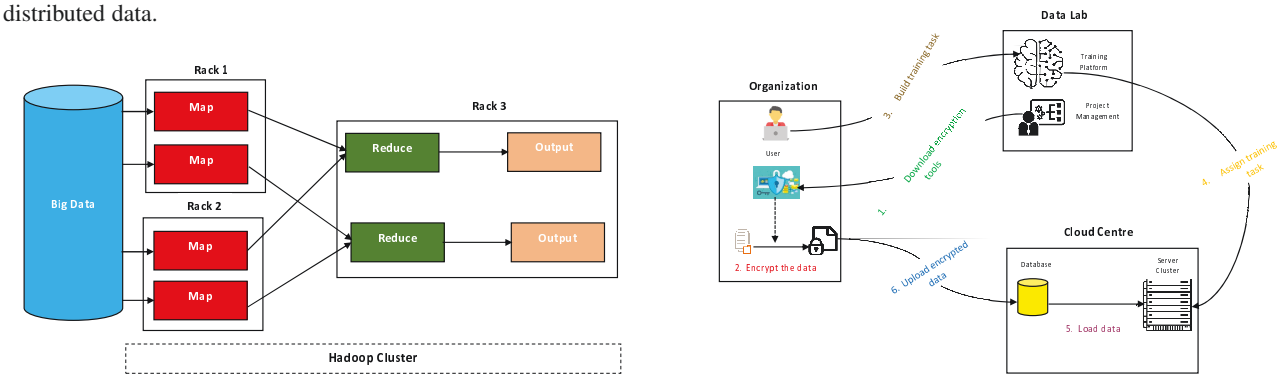
In this learning approach, the training dataset is distributed over multiple learners to be trained by different machine learning models. The results from the different learning models are then aggregated using some aggregation techniques (e.g., bagging, boosting, etc.) to improve the training accuracy. The objective of ensemble learning is to learn from a mixture of models to minimize the probability of relying on one insufficient model, rather than trying to improve a single global model using naturally distributed data as is the case in federated learning. Moreover, unlike federated learning wherein the data is non-Independent and Identically Distributed (non-IID) and depends on each single client's activities, the data in ensemble learning are split in an IID fashion as they come from one single training dataset.



(a) Distributed Machine Learning

(b) Ensemble Learning

Fig. 5: The main difference between distributed learning and federated learning is that in the former concept workers are not supposed to receive any global model from the server at each iteration, while in the latter workers receive a new version of the global model and derive updated parameters. The main difference between ensemble learning and federated learning is that the objective of the former is learn from a mixture of models to enrich the learning process, while in the latter the objective is to improve one single global model using naturally distributed data.



(a) MapReduce

(b) Shared Machine Learning

Fig. 6: The main difference between MapReduce and federated learning is that workers in the former concept are cluster machines that are deployed by the task owner, while in the latter workers are automatous and non-technical. The main difference between federated learning and shared machine learning is that the former only solves problems with in-domain data while the latter relaxes this restriction.

2.2.3 MapReduce and Parallel Learning

MapReduce [27] is a programming model for processing huge amounts of data over a cluster of machines. The MapReduce model consists of two phases, i.e., *Map* and *Reduce*. In the *Map* phase, the input data are processed to generate a set of intermediate key value pairs. In the *Reduce* phase, the intermediate values outputted in the *Map* phase and associated with the same intermediate key are combined and processed to produce the final results. MapReduce is logically related to federated learning. On can think of the parameter server in federated learning as the *Reducer* and the client devices as the *Mappers*. However, there is one essential difference between federated learning and MapReduce at the technical level. First, in federated learning the clients are autonomous and actually own the data that they analyze. Hence, they autonomously take the decision on whether or not to participate in the training. In MapReduce, on the other hand, *Mappers* are cluster machines that are owned by the data analytics task owner. This means that the availability of federated learning drastically varies based on the participation decision of the clients. On the other hand, the availability in MapReduce is more stable since the participation of the machines is guaranteed. The only availability burden in MapReduce would stem from some failure on the cluster machines. In addition, the data in MapReduce are split in an IID fashion, where a single training dataset is split into chunks (usually of 128MB) and stored in the Hadoop Distributed File System (HDFS) to be executed locally there by the *Mappers*.

2.2.4 Gossip Learning

Gossip learning [28] is a variation of federated learning in which no central entity is needed to initialize a global model or to aggregate model updates from local learners. Similar to federated learning, gossip learning is based on an on-device collaborative training architecture in which no sharing of the raw data needs to happen. Different from federated learning, gossip learning is fully decentralized in the sense that no parameter server is required. Instead, client devices directly share their model updates and the aggregation takes place in a distributed fashion.

2.2.5 Shared Machine Learning

Shared Machine Learning (SML) is a recent machine learning approach whose main goal is to achieve the aggregation of multiparty data while preserving the privacy of the involved individual users. It was introduced by Alibaba Subsidiary Ant Financial, one of the major financial companies in the World [6]. SML relies on one of the following data protection techniques, i.e., Trusted Execution Environment (TEE) and Multiparty Computation (MPC) systems. TTE is a processing environment that operates on a separation kernel[7] to provide resilience against tampering. It ensures the soundness of the executed code,

6. https://medium.com/syncedreview/shared-machine-learning-ant-financials-solution-for-data-privacy-8069cffe7bb6

7. The separation kernel is a security kernel that is employed to simulate a distributed system with the purpose of enabling the collocation of distinct systems demanding distinct levels of security on the same platform [29].

TABLE 3: Summary of the main commonalities and differences between federated learning and the other learning approaches

Approach	Characteristics
Distributed Learning	Learners train their models locally and send the derived parameters to the server. The server does not share any initial or updated training model with the workers.
Ensemble Learning	The training of a single dataset is distributed over a mixture of learners to be trained using different machine learning models. The data is split in an IID fashion since they come from one single dataset.
MapReduce	Executed in two phases: Map and Reduce. Learners (Mappers) are cluster machines owned by the data analytics task owner. The availability of MapReduce is almost stable.
Federated Learning	The server shares an initial and several updated models with learners. Autonomous learners train the shared model locally on their datasets and send the updated parameters to the server. The availability is variable and depends on the participation of autonomous learners. Data is split in an non-IID fashion since each learner's data depend this learner's own activities.
Gossip Learning	A fully decentralized federated learning approach in which clients directly share their model updates and aggregate them in a distributed fashion.
Shared Machine Learning	A recent machine learning approach whose main goal is to carry out the aggregation of multiparty data while preserving the privacy of the involved individual users.

the integrity of the runtime states (e.g., memory Input/Output CPU registers) and the privacy of the data, code and runtime states [30]. An SML algorithm that leverages TTE for data protection on multiparty distributed data usually follows the subsequent steps:

Users download encryption tools from a Data Lab[8], which employs the Registration Authority (RA) process to guarantee that the encrypted data can only be decrypted in the specified enclave[9];
Users encrypt their data using the encryption tool and upload the encrypted version to a cloud storage system;
Users construct training tasks on the training platform of the Data Lab;
The training platform sends the training tasks to a training engine, which fetches the encrypted data from the cloud storage to accomplish the desired training jobs.

On the other hand, MPC is an emerging cryptography approach that can jointly compute a function over data distributed across multiple parties, without breaching the privacy of the individual participants' data. MPC is widely used in conjunction with federated learning as discussed later in Section 7 2. An SML algorithm that leverages MPC for data protection on multiparty distributed data usually follows the subsequent steps:

Users download and locally deploy training services from the Data Lab;
Users construct training tasks on the training platform of the Data Lab;
The training platform sends the training tasks to a training engine, which in its turn sends the training task to the training server on the organization's side;
Workers of the organization load the data locally and execute the training jobs using multiparty security protocols.

The main difference between federated learning and SML is that the former can only operate with in-domain data and demands, as a prerequisite, that participating users have similar identities and statuses, which makes it compatible only with MPC for data protection. On the other hand, SML poses no restrictions on the identity of the users or on the type of the data, which makes it support both MPC and TTE.

2.3 Federated Learning and Emerging Communication Technologies

In this section, we explain how federated learning relates and can be capitalized on in several emerging technologies such as Internet of Things (IoT), cloud computing and edge computing.

2.3.1 Internet of things (IoT)

Machine learning has significantly contributed to the success of IoT in many aspects such as management, deployment, security and privacy, and data analytics and decision-making. The success of machine learning mainly stems from the abundance of big training data and enormous computation power [31], [32]. Nonetheless, the adoption of machine learning in IoT has always been hindered by privacy concerns. In fact, IoT data are often generated by IoT devices possessed by individuals who have serious privacy concerns and might not be willing to share their personal data. Moreover, the number of IoT devices is exponentially increasing (80 billion devices by 2025 as reported by the Intersectional Data Corporation), leading to a tremendous surge in the amounts of generated data (847 ZB of data will be generated at the network edge by 2021 as estimated by Cisco[10]). Hence, storing and processing all these fata in a centralized fashion becomes quite hard and inefficient. Federated learning has been lately proposed to answers these two concerns through enabling end devices (e.g., IoT devices) to cooperatively train machine learning models without having to share their data. As discussed in Section 2.1, the main idea of federated learning is to allow end devices to train local models on their local own data and then share only the model parameters (and not the raw data) with a parameter sever that aggregates the parameters of the different local models to come up with a global model. This concept has shown a great potential in reducing the latency of the training process and preserving the privacy of the data. On the other, IoT has also a lot to offer for federated learning especially in the coming years where IoT devices are expected to be empowered by $5G$ $6G$ networks with considerably high bandwidth and low latency. This would enable IoT devices to make efficient use of their computing resources to train their local models in an faster and better-performing way.

2.3.2 Cloud Computing

In traditional centralized data analytics scenarios, the data gathered by mobile devices (e.g., IoT devices, smartphones, etc.) are transmitted to a cloud-based server or data center to be processed in a centralized fashion. This approach, however, suffers from two fundamental limitations [15]. The first one relates to the privacy concerns of the data owners, which feel reluctant to share their own data with the cloud. This has led to the elaboration of strict data privacy legislations such as the Consumer Privacy Bill of Rights in the US and the European Commission's General Data Protection Regulation (GDPR). For example, the Articles 5 and 6 of the GDPR states that data collection and storage should be restricted to only what is user-consented and decidedly indispensable for processing. The second limitation of adopting a cloud-based architecture arises from the

8. A Data Lab is a Laboratory that is consecrated to do the experimentation and qualification of the company's data.

9. A specific part of a chip that is used to store sensitive information.

10. https://www.cisco.com/c/en/us/solutions/executive-perspectives/annual-internet-report/index.html

long communication latency, which originates from the long distance between the mobile devices that are highly geographically distributed and the cloud-based servers that are highly centralized in isolated areas.

Federated learning overcomes the privacy concerns that arise in the central cloud-based architecture through enabling an on-device collaborative training of a single machine learning model without having to share the raw training data with any cloud-based third-party entity. This is achieved through first initializing a global machine learning model on a central server over a few iterations to obtain some initial model parameters. These model parameters are then sent to a set of clients (data owners), which use their own resources to locally train a machine learning model with the shared parameters on their own data to derive an updated version of the parameters. Each client then sends its own updated parameters to the server, which aggregates the parameters from the different clients to produce a new global model. This process is repeated for several iterations until the global model reaches a certain desired accuracy level.

It overcomes the communication latency problem through eliminating the need to send raw data to a geographically distant cloud system and, instead, requiring that only model parameters be shared with (often) edge servers which are usually located in the proximity of the data owners. It is worth mentioning yet that communication is still a challenge in federated learning as model updates can be in the range of gigabytes in the current deep learning models, where millions of parameters might need to be shared with the edge servers at each iteration, for a non-negligible number of iterations. The communication aspect of federated learning is discussed in-depth in Section 4.

2.3.3 Edge Computing

The concept of *edge computing* or *edge intelligence* has been proposed to overcome the limitations of the cloud-based architecture when it comes to distributed data analytics. The main idea of edge computing is to capitalize on the storage and computing capabilities of edge servers and end devices to train the machine learning models closer to the origins of the data [33]. According to this concept, the training data are first communicated to the edge servers (or end devices) to carry out the low-level deep learning tasks [34]. Then, the intensive computation duties are offloaded to the cloud server which enjoys larger computation capabilities. Edge intelligence is a key enabler for several critical applications such as intensive care, ambient intelligence, industrial automation and oil/gas mining in deserted areas. For example, with edge intelligence, real-time data analytics and decision-making becomes possible for closed-loop applications in which some critical physiological parameters, such as blood glucose level or blood pressure, must be maintained within a specific range of values. Similarly, edge intelligence is perfectly suited for elder people's daily activity monitoring to promptly detect anomalies such as a fire or a fall and hence take quick measures by calling emergency.

A successful deployment of edge intelligence requires equipping the underlying (edge) devices with the following ingredients:

Connectivity: to enable the devices to connect to the network (e.g., Internet, local network) to exchange messages.
Computing: It is necessary to equip the devices with internal computing resources such as processing chips to enable data analytics in near real-time.
Controllability: The devices should be able to implement decisions, take actions, make prompt changes and incite actions across the network.
Autonomy: It is important for the devices to be supplied with autonomous computing capabilities that enable them to monitor and manage their own data and resources by their own.

In real-world scenarios, edge devices are asked to execute deep learning algorithms that employ multiple layers on the inputs prior to delivering the output. This is because edge devices are mostly used to run image and speech recognition, anomaly detection and natural language processing. Executing deep learning algorithms typically involves large parallel matrix multiplication operations, thus requiring specialized hardware. To make this feasible, edge intelligence capitalizes on several rising technologies such as Visual Processing Units (VPUs) and RISC-V. A VPU is an emerging category of microprocessor that is tailored for artificial intelligence tasks. It aims to accelerate the execution of machine vision algorithms such as Convolutional Neural Networks (CNNs) and Scale-Invariant Feature Transform (SIFT). VPUs are designed to include direct interfaces to fetch data from cameras without undergoing any off-chip buffers and to support more on-chip dataflow between several parallel execution units. On the other hand, RISC-V is an Instruction Set Architecture (ISA) that embodies the vision of the Reduced Instruction Set Computer (RISC) standards. RISC-V is designed to be open-source and free of charge. Unlike most other ISA designs such as x86 or ARM, RISC-V is designed in such a way that the instruction set can be customized to support specialized and extreme application demands such as artificial intelligence and machine learning. This is helpful, for example, in a camera surveillance scenario to decrease the amount of video frames that need to be transmitted to the cloud through crafting the application's design and specialized processor in such a way that most video frames could be handled by end or edge devices and only a small fraction of the frames (e.g., those that exhibit some considerable changes) go to the cloud for further analysis. We summarize in Table 4 the hardware specifications of some known existing edge devices to give the reader an idea of how edge devices should be configured to support federated learning applications[11].

When coupled with federated learning, edge intelligence brings along many non-negligible benefits [15], [35], [36], which we discuss hereafter:

Reduced Latency and Bandwidth Utilization: Edge intelligence offers the flexibility to partition the deep neural networks in such a way that some layers could be processed on the edge devices, while the rest can be processed in the cloud. In other words, the initial layers of the neural network which are generally feature-abstraction operations can be executed at the level of the edge devices. Then, as the information propagates through the neural network, the layers get abstracted into high-level features which are quite less heavy than the initial ones. These layers can then be transmitted to be processed by the cloud with minimal bandwidth consumption and latency. Such an approach is perfect to be adopted with IoT devices that rely on communication technologies that support limited payload size such as Long Range (LoRa) and Narrowband Internet of Things (NB-IoT) [37].
Security: Edge intelligence introduces some interesting security benefits to the federated learning process when compared to the traditional cloud-based architecture. In fact, the fully centralized nature of cloud computing makes the federated learning particularly vulnerable to distributed denial of service (DDoS) attacks and power outages. On the other hand, the fact that edge computing divides the storage and processing over a large set of devices and data centers complicates the job of attackers in successfully causing any single interruption or network shutdown. Moreover, as most of the data are processed on local/edge devices instead of being sent to a central cloud centre, edge computing decreases the risks of crafting attacks on these data. In other words, there is less data

11. https://www.therobotreport.com/why-and-how-to-run-machine-learning-algorithms-on-edge-devices/

TABLE 4: Edge Device Hardware Specifications for Federated Learning

Edge Device	GPU/VPU	CPU	Machine Learning Software Support
Raspberry Pi	VideoCore VC6	Quad ARM Cortex-A72	TensorFlow and TensorFlow Lite
Coral System-on-Module (SoM) – Google	Vivante GC7000Lite	Quad ARM Cortex-A53 + Cortex-M4F	TensorFlow Lite and AutoML Vision Edge
NVIDIA Jetson TX2	NVIDIA Pascal	Dual Denver 2 64-bit + quad ARM A57	TensorFlow and Caffe
Intel Neural Compute Stick 2 (NCS2)	Movidius Myriad X VPU		TensorFlow, Caffe and OpenVINO toolkit
RISC-V GAP8			TensorFlow
ECM3531 A – Eta Compute	ARM Cortex-M3 + NXP CoolFlux DSP		TensorFlow and Caffe
ARM Ethos N-77	8 NPUs in cluster, 64 NPUs in mesh + NXP CoolFlux DSP		TensorFlow, TensorFlow Lite, Caffe2, PyTorch, MXNet and ONNX

to be intercepted during transmission, and even if some edge devices get compromised, only the data that is gathered by and contained in these devices will be at risk, compared to the whole set of data that could be intercepted on a compromised cloud server. Yet, it is unfair not to mention that the distributed nature of edge intelligence triggers some security risks where, for example, one edge device might be employed as a point of entry to carry out cyber-attacks, enabling the attack to infect a whole network from a single weak point. Nonetheless, the good news is that the same distributed nature of edge intelligence that makes room for such a type of attacks makes it also easier to craft security solutions that can quarantine a fraction of infected edge devices without having to shut down the whole network.

Scalability: Companies (i.e., federated learning task owners) are expected to be always growing in terms of exposure and also in terms of infrastructure needs. Yet, predicting the exact needs in terms of infrastructure is not always possible. These needs typically include constructing a dedicated and expensive data center with all the essential complex Information Technology (IT) equipment. Such private and expensive facilities put strict limits on the potential growth of the companies, locking them into their rigorous forecasts. However, in case the actual growth overrides the company' expectations, these companies might have no choice but ceding those additional opportunities due to insufficient resources. Edge intelligence can be capitalized on to facilitate the business scaling of the companies. Moving the storage, analytics and computing abilities to edge devices that have smaller footprints and can be deployed in the proximity of end users enables companies to connivently extend the reach and capabilities of their edge network while avoiding the burdens of building complex centralized data centers. Edge intelligence along with the related concept of *Colocation*[12] enables companies to easily and rapidly extend their edge network's reach and cost-effectively.

Reliability: By enabling the machine learning to be carried closer to end users, edge computing reduces the chance of a network problem in a distant data center influencing local users' experience. Even if some nearby regional data center experiences an outage, due to the autonomy property discussed earlier, the edge devices will be able to effectively pursue their operations with their native processing resources. Having a multitude of edge data centers and edge computing devices makes it more unlikely for a single failure to entirely disrupt the whole service. Thus, several pathways are available to reroute the exchanged model updates to in case of any failure to maintain users' access to their federated learning services.

Transfer Learning Empowerment: As previously discussed, in an edge intelligence scenario, the execution of the deep neural networks is distributed among the edge devices and the cloud. This unique opportunity can be capitalized on to re-purpose the neural network for a completely different application by only tuning the layers that are executed at the level of the cloud, without having to change the layers at the edge devices. Knowing that the application logic of the layers that are in the cloud is reasonably easy to tune, this unique architecture gives the opportunity to use the same devices to model different applications. The practice of tuning parts of the machine learning model to carry out different applications is a concrete example of *Transfer Learning*, which is explained in more details in Section 3.1.2.

Data Regeneration: Provided that the feature abstraction duties are executed at the level of the edge devices, it becomes possible to reconstruct an approximation of the original data at the level of these devices themselves without having to get large inputs from the cloud. This further reduces the bandwidth utilization and the latency of the data regeneration and hence the overall federated learning process.

12. The practice of renting the space and computing hardware (e.g., networking, power and cooling components, physical security, etc.) provided to the edge devices at a third-party provider's data center facility.

2.3.4 5G/6G Networks

The telecommunications industry continues to evolve over the years, providing users with increased connection speeds and improved networking experience. Numerous cellular network versions have been unleashed over the past years to meet the augmented needs of users for speed and capacity. Currently, the Fourth-Generation (4G) is still the most widespread generation of cellular communications. From a technical perspective, the 4G provides Internet speeds that are approximately ten times faster than they were in the Third-Generation (3G). Similar to the 3G, the 4G technology can be divided into two broad components, i.e., Long-Term Evolution (LTE) and Worldwide Interoperability for Microwave Access (WiMax). Choosing between LTE and WiMax mainly depends on the option that is available to the users and how they intend to use it. WiMax is generally a better option in case users need a connectivity for a fixed location and are situated not far from the source of this connectivity. On the other hand, LTE is more suitable for situations wherein users need connectivity on the go in a broad range of locations, owing to the wide coverage and enhanced consistency provided by this type of networks. Beyond the 4G, the upcoming Fifth-Generation (5G) networks promise to uphold a wider spectrum of services such as Augmented Reality, Virtual Reality, large-scale IoT, and autonomous driving. This will be enabled though the three essential technical characteristics this generation of networks, i.e., enhanced Mobile BroadBand (eMBB), Ultra-Reliable Low-Latency Communications (URLLC), and massive Machine-Type-Communications (mMTC). By providing throughput speeds of up to 20 Gbps, eMBB is mainly tailored to support new data-driven use cases that demand high data rates such as content sharing in stadiums and arenas, smart cities and unlimited virtual experiences. URLLC provides ultra-responsive connections with ultra-low latency. Contrary to eMBB, data rates in URLLC are not intended to be very high but the connection is designed to support high mobility. This makes URLLC suitable for mission-critical applications such as remote medical assistance, autonomous driving and industrial automation. The main focus of mMTC is to offer connectivity to a large number of devices, yet with low reliability. It can offer long-range communication with energy efficiency and asynchronous access, features that are suitable for low-power devices in an enormous quantity such as IoT devices. In short, the purpose of the 5G networks is to move the wireless communication world from a communication-

oriented architecture to a service-based architecture that supports the idea of "connected things".

The Sixth-Generation (6G) is the successor of the 5G which promises to achieve a transition from the concept of "connected things" offered by the 5G to that of "connected intelligence" [38]. In fact, the 6G is expected to rely on advanced artificial intelligence foundations to craft innovative and intelligent services through efficiently collecting, communicating and analyzing data anytime, anywhere. To fulfill this promise, the 6G needs to be paired with the concept of ubiquitous AI, which is interested in making the different aspects of networking human-centric instead of being data-, machine-, or application-centric. Achieving the dream of ubiquitous AI is however not possible with the traditional machine learning design which is based on a central cloud-based server architecture. In fact, the privacy considerations and communication resource limitations in future intelligent wireless networks make it unpractical to let the different wireless devices that participate in the machine learning scenario to send all their collected data to the cloud for centralized analytics. Therefore, the academy and industry push is increasingly shifting toward the design of decentralized machine learning frameworks that enable wireless devices to execute a shared learning model without having to transmit their local data. Federated learning stays at the core of this trend, given all the revolutionary benefits it promises to bring to the machine learning experience.

The relationship between federated learning and 5G/6G networks is bidirectional in the sense that federated learning offers many benefits toward boosting the adoption of 5G/6G networks and 5G/6G are also great contributors to the success of the federated learning experience. In fact, federated learning is a perfect candidate for solving many challenges in 5G/6G networks. For example, federated reinforcement learning algorithms can be used to provide efficient solutions to complex convex and non-convex optimization problems that could be designed to model several crucial problems such as resource management, network control, interference alignment and user grouping [39]. Moreover, federated supervised learning algorithms can be used to provide a wide range of analytical services in these rising networks, including but not limited to wireless environment analysis, user identifications, user authentication, access control management, behavior prediction, and intrusion detection and prevention. On the other hand, 5G/6G networks could also be capitalized on to foster the applicability and efficiency of federated learning. Specifically, by equipping the IoT and edge devices with 5G/6G networks that enjoy high bandwidth rates and low latencies, these devices will be able to make more efficient use of their computing resources to train their local models in an faster and better-performing way. Likewise, the processes of exchanging global model and model updates between the client devices and parameter server will be more efficient thanks to the advantages brought by the eMBB technology which is integrated into the design of both the 5G/6G networks.

2.3.5 Applications of Federated Learning in Networking

In the domain of networking, federated learning has mainly been used for task scheduling and resource allocation. It is particularly useful in these scenarios as it enables the decentralization of the decision-making process and the modularization of the task that each network node would be responsible for. In the following, we survey and discuss in more detail the main approaches that employed federated learning to solve challenges related to the networking domain.

For example, the authors of [40] and [41] capitalize on federated learning to enable ultra-reliable and low-latency vehicular communications. To do so, a network-wide power minimization problem is formulated to assure high reliability and low latency in terms of probabilistic queueing delay. First, statistics of queue delays exceeding a certain threshold are acquired using the extreme value theory. A distributed maximum likelihood estimation technique based on federated learning is then advocated. Each vehicle gradually (over different time scales) constructs and shares a local model consisting of Generalized Pareto Distribution (GPD) parameters (i.e., scale and shape used to the parts of the queue delay distribution that are quite far away from the mean) to the roadside unit, which is then responsible for aggregating the models of the different vehicles. Thereafter, Lyapunov optimization is employed to solve the optimization problem and determine the appropriate resource allocation strategies. The authors of [42] employ federated learning to improve the offloading decision-making in Internet of Things (IoT) environments. A distributed Deep Reinforcement Learning (DRL) approach is implemented over edge devices to decide on whether a certain task should be executed locally on the IoT devices or offloaded to the edge nodes. In the DRL model, the network states are modeled as a function of task execution makespan, energy queue length, channel gain between the IoT device and corresponding edge nodes, and task handover delay. The reward function is modeled with regards to task queuing delay, task execution delay, penalty of execution failure and number of failed tasks. The DRL is implemented based on federated learning, where at each round, a collection of IoT devices is randomly chosen to download the DRL model parameters from the edge network. IoT devices train the model on their own data consisting of local sensing data, available energy level and channel gain. They then transmit the updated model back to the edge nodes for global aggregation. In [43], the authors investigate the effectiveness of federated learning for image classification in Vehicular Edge Computing (VEC). A selective model aggregation approach is proposed, where only good-quality local deep neural network models are chosen to be sent and aggregated by the central server. Since the server has no a priori knowledge about the image quality and computation capabilities of the vehicles, the selection problem is modeled as a two-dimension contract theory model between the server and vehicle clients. The problem is then converted into a tractable problem to be solved by a greedy algorithm through relaxing some of its constraints. In [44], the authors discuss a communication-efficient hierarchical federated learning approach in which mobile users with local datasets are grouped around small-cell base stations to carry out the federated training in a decentralized fashion. The base stations then periodically communicate with a macro-cell base station, which aggregates the model updates from the different base stations to construct a shared global model. In [45], the authors address the problem of noise that hinders the wireless communications in the broadcast and aggregation processes of federated learning. The problem is formulated as a parallel optimization problem under the worst-case and expectation-based models. The optimization problem that is modeled under the expectation-based model has been solved using a loss function approximation algorithm, whereas the optimization problem that is modeled under the worst-case model has been solved using a successive convex approximation algorithm. The authors of [46] propose a novel architecture for the Internet of Vehicles (IoV) that capitalizes on federated learning to alleviate the transmission overhead and address the providers' privacy concerns. Specifically, a hybrid blockchain model that consists of a two-stage verification process is advanced to address the privacy concerns. Moreover, an asynchronous federated learning approach that uses deep reinforcement learning for vehicle selection is advanced to improve the efficiency. In [47], the authors investigate the impact of the random scheduling (RS), round robin (RR), and proportional fair (PF) scheduling strategies on the performance of federated learning in large-scale wireless networks. The access points and wireless user equipment locations are deployed following an independent Poisson point process. The results suggest that running federated learning with PF achieves a better performance compared to that of RS and RR in case the network is running under a high Signal-to-Interference-Plus-Noise Ratio (SINR) threshold. Conversely, in case the network operates under a low SINR, the RR scheduling strategy would be more

preferable.

Recently, federated learning has been studied to address challenges in Unmanned aerial vehicle (UAV) networks [48], [49]. For example, in [50], the authors adopt federated learning to enable privacy-preserving machine learning across a group of autonomous Drones-as-a-Service (DaaS). The objective is to boost intelligent transportation systems applications such as car parking management and traffic prediction. Moreover, the authors put forward a contract theory-based approach to guarantee that UAVs truthfully report their types. In [51], the authors combine federated learning with mean field game theory to achieve the control of large-scale UAVs, while reducing the inter-UAV communication overhead. In [52], the authors capitalize on federated learning to design aerial-ground air quality sensing framework for fine-grained 3D air quality monitoring and prediction. In [53], the authors address the problem of intermittent connections among UAV swarms and ground base stations, which hinders the adoption of centralized machine learning for executing various tasks (e.g., target recognition, etc.). To do so, a federated learning approach is proposed where each UAV trains a local federated learning model based on its collected data and then forwards the trained model to a leading UAV node, which is responsible for model aggregation. In [54], the authors propose to capitalize on coalitions of UAVs as wireless relays to facilitate and reduce the cost of communications between the Internet of Vehicles (IoV) nodes and federated learning server. To form coalitions, a joint auction-coalition formation approach is proposed to assign UAV coalitions to the groups of IoV components, in such way to maximize the sum of UAVs individual profits.

In addition to the existing approaches, federated learning can be employed to address several other aspects of networking, a few of which are subsequently explained. The first aspect is that of content caching. The main idea of content caching is to bring the popular content closer to the edge terminal so that it can be efficiently accessed locally and delivered to end users. The building block of this approach is to analyze users' activities to determine the popular content. In this regard, federated learning can be used to predict popular content on mobile devices without having to directly access users' data. The second aspect is that of spectrum management. In this context, federated learning can be used to devise effective spectrum access decision-making models. This can be done through representing each radio as a separate client that communicates its local utilization data to a central entity, which then explores these data to build global spectrum access prediction models. The third aspect is that of core networking duties. In this context, federated learning can be of prime utility to modularize the duties of each network node and hence minimize the overhead. This can be achieved through assigning each node a little number of specific (modular) tasks (e.g., session management, access mobility management, etc.) and then employing vertical federated learning to aggregate the results and take come up with efficient decision-making models.

We summarize in Table 5 the main approaches that employ federated learning to solve networking challenge.

3 Statistical Challenges (Challenge 1)

The statistical challenges in federated learning can be classified into four essential sub-challenges: (1) non Independent and Identically Distributed (non-IID) Data; (2) block cycles; (3) model heterogeneity; and (4) bias mitigation. We discuss each of these sub-challenges in detail and provide a classification of the techniques that are used to address each particular sub-challenge. Most of the approaches that address statistical challenges are interested in solving problems related to non-IID data. Because of the abundance of approaches that tackle challenges related to non-IID data and for the reader's convenience, we provide a sub-classification of the non-IID challenge based on the type of imbalance taken into consideration, resulting in three classes, i.e., class imbalance, distribution imbalance and size imbalance. The classification scheme of the statistical approaches is schematized in Fig. 7. Moreover, we provide in Table 6 a summary of the main approaches that tackle statistical challenges in federated learning and highlight the criteria (proposed in Section 3.5) that each underlying approach satisfies.

3.1 Non Independent and Identically Distributed Data

The non-IID data challenge arises because of the inherent heterogeneity in the local data generated across clients' device. Since each local device records the activities of its owner, data across devices tend to have different sizes, features, and target classes distribution. This technically means that the local data of one single client cannot be considered to be representative of the overall data distribution [78]. Three scenarios of non-IID data in federated learning can be encountered, i.e., class imbalance, distribution imbalance and size imbalance. In the following sections, we explain each of these scenarios in detail, provide a classification of the techniques used to address each underlying scenario, and discuss in detail the existing approaches proposed under each particular technique.

3.1.1 Class Imbalance

This type of imbalance occurs when the total number of instances classified to fall under a certain target class (e.g., non-attack, non-disease, etc.) is far higher than those classified to fall under another class (e.g., is-attack, is-disease, etc.). Although this problem is common in machine learning, i.e., in single centralized large datasets, it becomes even more widespread in federated learning due to the geographical distribution of the clients. For example, consider a mobile keyboard emoji prediction scenario using the federated learning framework, where the data is split across a large number of mobile users spread across the World. In such a scenario, some emojis might be banned or less used in some geographical areas. For instance, Apple recently decided to hide Taiwan's flag emoji for users that are based in Hong Kong or Macau[13]. This results in a considerable class imbalance, where the number of instances associated with Taiwan's flag emoji is expected to be much less than other emojis. Two main techniques are used to address the class imbalance challenge, i.e., *data augmentation* and *active learning*. In the following, we discuss each of these techniques in detail and shed light on the approaches that employ these techniques.

Data Augmentation: The main idea of this technique consists in augmenting clients' local data with thoughtfully designed additional data in such a way to make the global distribution of data over the different target classes more balanced. For example, the authors of [55] propose a self-balancing federated learning framework called *Astraea* to address the class imbalance problem. The proposed framework is composed of two modules, i.e., data augmentation and multi-client scheduling. In the data augmentation module, the aggregation server computes the amount of augmentation needed for each class according to the global distribution of the data. It then assigns the augmentation tasks to the clients to be done in parallel. The augmentation process takes as input one sample and outputs the augmentations which include random rotation, random zoom, random shear and random shift of the sample. In the multi-client scheduling module, a greedy technique is employed to allow each mediator (controlling a set of clients) to go through the data distribution of all unsolicited clients (i.e., clients that weren't asked to participate in the federated learning process) and pick out those clients whose data distributions would make the mediator's overall data distribution to be the closest to a uniform distribution. In [56], the authors first show, using experiments

13. https://www.theverge.com/2019/10/7/20903613/apple-hiding-taiwan-flag-emoji-hong-kong-macau-china

TABLE 5: Summary of the federated learning application domains

Approach	Application Domain	Main Idea
Samarakoon et al. [40], [41]	Networking	A distributed maximum likelihood estimation technique based on federated learning wherein each vehicle constructs and shares a local model with the roadside unit, which is then responsible for aggregating the models of the different vehicles.
Ren et al. [42]	Networking	A federated learning-based deep reinforcement learning approach over edge devices to decide on whether a certain task should be executed locally on the IoT devices or offloaded to the edge nodes.
Ye et al. [43]	Networking	A federated learning approach for image classification in vehicular edge computing in which only good-quality local deep neural network models are chosen to be sent and aggregated by the central server.
Abad et al. [44]	Networking	A hierarchical federated learning approach in which mobile users with local datasets are grouped around small-cell base stations to carry out the federated training in a decentralized fashion.
Ang et al. [45]	Networking	A parallel optimization model to address the problem of noise that perturbates the wireless communications in the broadcast and aggregation processes of federated learning.
Lu et al. [46]	Networking	A novel architecture for the Internet of Vehicles (IoV) that capitalizes on federated learning to alleviate the transmission overhead and address the providers' privacy concerns.
Yang et al. [47]	Networking	A analytical framework to investigate the impact of the random scheduling (RS), round robin (RR), and proportional fair (PF) scheduling strategies on the performance of federated learning in large-scale wireless networks.
Lim et al. [50]	Networking	A federated learning-based approach to enable privacy-preserving machine learning across a group of autonomous Drones-as-a-Service (DaaS).
Shiri et al. [51]	Networking	An approach that combines federated learning with mean field game theory to achieve effective control of large-scale UAVs, while reducing the inter-UAV communication overhead.
Liu et al. [52]	Networking	A federated learning-based aerial-ground air quality sensing framework for fine-grained 3D air quality monitoring and prediction.
Zeng et al. [53]	Networking	A federated learning approach to address the limitations of centralized machine learning in situations where the connections among UAV swarms and ground base stations could be intermittent.
Ng et al. [54]	Networking	Coalition formation scheme of UAVs to serve as wireless relays for facilitating and reducing the cost of communications between the Internet of Vehicles (IoV) nodes and federated learning server.

performed on a neural network, that the accuracy of federated learning decreases by up to 55% on highly skewed non-IID where each client exclusively trains a single class of data. This decrease is attributed to the weight divergence, which practically represents the distance between the probability distributions over classes on each device and the probability distributions of the population's distribution. To overcome this problem, the authors propose to create a small dataset and share it with all the clients. The local model of each client is then trained on both that shared data and the private data of that client. Experimentally, the authors show that with only 5% of globally shared data, the accuracy of the neural network trained in a federated fashion increases by approx. 30% using the CIFAR-10 dataset. In [57], the authors point out the difficulty of finding publicly available IID large datasets as proposed in [56] and suggest to build a large IID dataset that is collected from a small number of clients that agree to share their data. This dataset is used by the server to generate model updates, which are then aggregated with the model updates trained on non-IID clients' data. The authors argue that adding model updates from an IID dataset helps improve the performance of the federated training process. The resource constraints on client device are also taken in consideration in the solution where an optimization problem that accounts for both data distribution across clients and channel condition of each client is designed and then solved using heuristics. In [58], a federated augmentation approach is proposed. In this approach, each device reports to a high-computing server the labels that are missing in its data samples and uploads some seed data samples of these labels. The server then trains a Generative Adversarial Network (GAN) to oversample the uploaded samples. Finally, each device downloads the trained GAN generator to supplement the missing labels, until attaining an IID training dataset. To ensure the privacy of the augmentation process, the authors propose to let each device to upload supplementary redundant data samples pertaining to labels other than those that are actually missing.

In what follows, we discuss the existing data augmentation approaches in federated learning vis-à-vis the conventional data augmentation methods to better position these approaches. The four common data augmentation methods in the literature can be classified into four major classes, i.e., *data warping*, *over-sampling*, *geometric transformations* and *Generative Adverserial Networks (GANs)*. The main idea of *data warping* is to create model character deformations that mimic the distortions that might occur in handwriting or printing. Distorted character data is used to make the amount of training examples for each class more balanced, thus reducing the classification bias. On the other hand, the main purpose of *over-sampling* is to address the problem of class imbalance in real-world datasets that comprise a small percentage of target class samples. Thus, synthetic

TABLE 6: Summary of the approaches that tackle statistical challenges

Approach	Challenge	Technique	Main Idea	Criteria
Duan et al. [55]	Class Imbalance and Size Imbalance	Data Augmentation	Determine the amount of augmentation needed for each class according to the global distribution of the data.	Criteria #5 and #7
Zhao et al. [56]	Class Imbalance and Size Imbalance	Data Augmentation	Create a small dataset and share it with the clients, where each client performs the training on both the shared data and private data.	Criterion #5
Yoshida et al. [57]	Class Imbalance and Size Imbalance	Data Augmentation	Build a large IID dataset through collecting some data from a small number of clients. The server uses this dataset to generate model updates, which are then aggregated with the model updates trained on the clients' non-IID data.	Criteria #5 and #9
Jeong et al. [58]	Class Imbalance	Data Augmentation	Each device reports to the server the labels that are missing in its dataset and uploads some seed data samples of these labels. The server trains a GAN to oversample the uploaded samples and sends it to each client to supplement the missing labels.	Criterion #5
Goetz et al. [59]	Class Imbalance	Active Learning	Evaluate a utility function on each client to quantify the loss of its local data and then accordingly favor the clients that considerably have higher loss as the ones having observations associated with minority class data.	Criterion #7
Smith et al. [60]	Distribution Imbalance	Multi-Task Learning	Enable each client to learn many separate but related models using MTL.	Criterion #9
Caldas et al. [61]	Distribution Imbalance	Multi-Task Learning	Embed non-linear mappings into the design of multi-task federated learning to capture non-linear relationships in the local training models and in the relationships between them.	Criterion #2 and #9
Sattler et al. [62]	Distribution Imbalance	Client Clustering	Cluster the clients based on the geometric properties of the federated learning loss surface.	Criteria #2 and #8
Ghosh et al. [63]	Distribution Imbalance	Client Clustering	A Lloyd clustering algorithm that employs $_2$-distance to group clients' data.	Criteria #2 and #8
Briggs et al. [64]	Distribution Imbalance	Client Clustering	A hierarchical clustering mechanism to group the clients based on the similarity of their local updates to the global joint model.	Criteria #2 and #8
Liu et al. [65]	Distribution Imbalance	Transfer Learning	A federated transfer learning approach in which a model is learned from a source-domain's data distribution to be applied on a different (yet related) target-domain's data distribution with high accuracy.	Criterion #12
Sharma et al. [66]	Distribution Imbalance	Transfer Learning	A federated transfer learning approach in the presence of malicious parties that arbitrarily deviate from the federated training process.	Criterion #12
Liu et al. [67]	Distribution Imbalance	Parameter Tuning	A momentum federated learning model that considers the last iteration along with the current gradient to accelerate the convergence of the federated training.	Criteria #1 and #6
Jiang et al. [68]	Distribution Imbalance	Parameter Tuning	A consensus-based distributed momentum SGD method that jointly satisfies the decentralized computation, data parallelization and constrained communication features.	Criterion #1
Koskela et al. [69]	Distribution Imbalance	Parameter Tuning	A learning rate adaptation mechanism using the moments accountant technique in an attempt to make the global model optimal for each client's local data.	None
Li et al. [21]	Distribution Imbalance	Parameter Tuning	Add a proximal term to the local subproblem of each client to restrict the influence of local updates through forcing them to be closer to the initial global model.	Criterion #6
Chen et al. [70]	Distribution Imbalance	Parameter Tuning	A gradient correction approach that perturbs the local gradients with unimodal and symmetric noise to push the expected median to draw near the expected mean of the gradients upon aggregation.	None
Wang et al. [24]	Distribution Imbalance	Parameter Tuning	A layer-wise federated learning model that considers the permutation invariance of the neurons prior to aggregation, thus allowing for global model size adaptation.	Criterion #6
Mostafa et al. [71]	Distribution Imbalance	Parameter Tuning	An adaptive online hyperparameter tuning strategy that relies on reinforcement learning to punish divergent representations across clients using a regularization term.	None
Ruan et al. [72]	Size Imbalance	Flexible Client Participation	Relax some of the restrictions on devices' participation to attract a larger number of clients having data that match those that are held by clients having less volumes of data than the rest.	Criterion #9
Wang et al. [73]	Size Imbalance	Flexible Client Participation	A deep Q-learning mechanism to thoughtfully choose, in each iteration, the subset of devices that maximizes the reward in terms of size balance and penalizes having more communication rounds.	Criterion #7
Eichner et al. [74]	Block Cycles	Plurality	A pluralistic solution whose main idea is to train a different model for each block in the cycle.	Criterion #3
Li et al. [75]	Model Heterogeneity	Knowledge Distillation	A knowledge distillation technique that enables clients to share model updates in a blackbox manner in terms of class scores on samples from a public dataset.	Criteria #5 and #10
Mohri et al. [76]	Bias Mitigation	Weighted Optimization	An agnostic federated learning approach in which the shared learning model is optimized for any target distribution consisting of a mixture of client distributions.	Criterion #11
Li et al. [77]	Bias Mitigation	Weighted Optimization	An optimization problem that boosts the fairness of the accuracy distribution across devices through assigning higher weights to the devices that generate higher loss.	Criteria #6 and #11

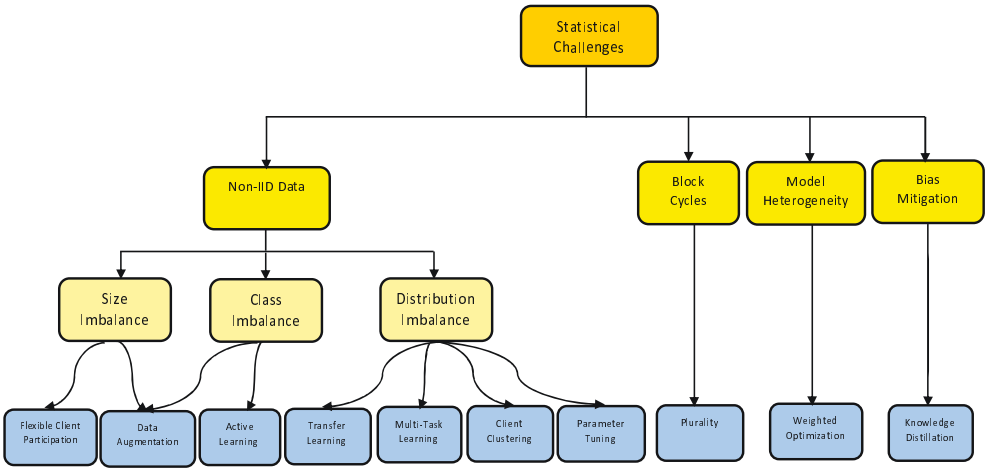


Fig. 7: Classification of the statistical approaches in federated learning

data samples are created in the feature-space from randomly selected pairs of real world feature samples from the minority classes. The main intuition is that, according to [79], higher level representations broaden the relative volume of plausible data samples within the feature space, thus narrowing the space for undesired data samples. To generate the synthetic data, simple transformations such as noise injection, interpolation, or extrapolation are applied to feature-space data. Similar to *data warping*, the method of *over-sampling* is based on modifying the real-world data samples to generate augmented data. Different from *data warping*, *over-sampling* has the advantage of being application-independent, given that it produces the synthetic samples in the learned feature-space.

The third data augmentation method is that of geometric transformations which includes rotation, flips, brightening, clipping, cropping and channel alterations. This type of data augmentation can only be applied to image datasets and generates data that are mere transformations of the original ones. The fourth and most recent data augmentation method is that of GANs. A GAN is composed of two components that are trained against each other, i.e., the *generator* and the *discriminator*. The generator is trained using a random noise vector to generate realistic synthetic data, in an attempt to trick the discriminator and make it unable to recognize that such data are generated and not real. On the other hand, the discriminator takes both the real and synthetic data samples as inputs to the training process, in order to be able to differentiate between the real and generated data samples. It outputs a probability value that quantifies the degree that certain samples come from possible data sources. The models are trained together in a zero-sum spirit, so that the improvements in the discriminator come at the cost of a reduced capability of the generator, and vice versa.

Most of the existing data augmentation techniques in federated learning [55], [56], [57] adopt the over-sampling approach in the sense that they are based on the idea of generating synthetic data in the feature-space from randomly selected pairs of real-world feature samples belonging to the minority classes. GANs have received less attention so far with one approach [58] having investigated their efficiency in minimizing the class and size imbalance in federated learning. Yet, we argue that GANs have a great potential to improve the quality of the synthetic augmentation data when applied in a federated learning environment.

Active Learning: Active learning is a special type of machine learning that is based on the assumption that the learning algorithm can perform better if it can choose the data it wants to learn from. Active learning allows the algorithm to actively query the programmer or a labeled dataset to learn the correct prediction for a given problem. Traditionally, *passive learning* methods are based on collecting large volumes of data that are randomly sampled from a certain distribution to be used for prediction. This task, however, is quite time-consuming especially when it comes to bringing labelled data which often are hard to obtain. To better clarify the idea of active learning, consider that you are given the task of predicting the mortality rate among Coronavirus patients. To do so, unfortunately, you might only have the chance to give a small number of patients further examinations to collect features. In other words, every day millions of people die across the World. Knowing exactly the proportion of people that are actually infected with Coronavirus out of these millions is almost impossible, due to the lack of testing facilities and personnel. In this case, instead of randomly choosing patients, you can choose patients based on some criteria, e.g., whether the person suffers from a chronic disease and is aged over 60. Determining such criteria can be done in a dynamic fashion. For example, if you observe that the model performs well at predicting the mortality rate for people over 60 but struggles to accurately predict for those aged between 45 and 60, then age might be a suitable criteria. This process of choosing instances based on some criteria is called *active learning* and is depicted in Fig. 8. In federated learning, active learning has been used to allow the federated training algorithm to choose the clients it wants to learn from in such a way to favor the ones having observations associated with minority class data. The objective is to make the overall data across the selected clients more class-balanced.

For example, the authors of [59] propose an active federated learning approach in which the clients, at each training iteration, are chosen according to a probability which takes into consideration the current model and the data available on the client's device. In more detail, a valuation function that is assessed on each client's device and then returned to the server is proposed. The valuation function is quantified in terms of loss. This way, drastic class imbalances and weak separation of classes could be detected as minority class data observations and will have considerably higher loss than those of majority class data observations. Consequently, the server would favor clients having more minority data points. In case all data points are equally valuable based on the valuation function, clients with higher volumes of data are given a higher preference.

3.1.2 Distribution Imbalance

This type of imbalance occurs when the feature distribution significantly varies across clients. In federated learning, different clients have distinct interests and activities, which means that the features that are available in one client's local dataset considerably vary from those contained in the rest of the clients' datasets. For example,

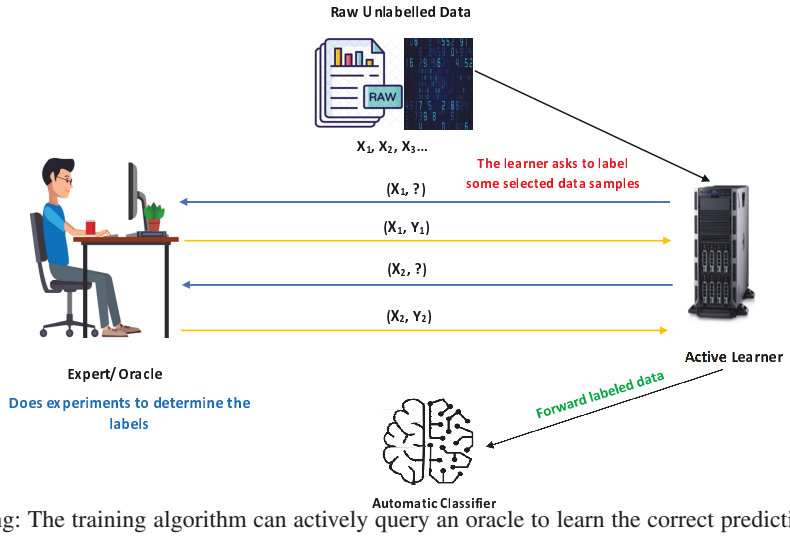
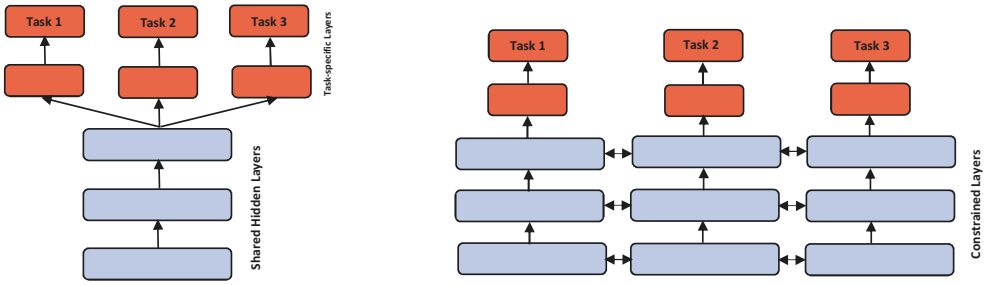


Fig. 8: Active Learning: The training algorithm can actively query an oracle to learn the correct prediction for a given problem.



(a) Multi-Task Learning with Hard Parameter Sharing Method

(b) Multi-Task Learning with Soft Parameter Sharing Method

Fig. 9: Multi-Task Learning: Several related tasks are trained together and the representations are sharing among them.

consider a federated training task where the purpose is to predict the attractiveness of human faces using data from different clients. In this scenario, clients will have varying tastes on the attractiveness of certain profiles, e.g., one half of the clients might think that bald men are attractive while the other half would think that they are unattractive. In such a scenario, it would be impossible for a single model to predict the attractiveness of bald men for all clients at the same time. Technically speaking, distribution imbalance is problematic since federated learning capitalizes on the stochastic gradient descent method, the most used method for training deep neural networks. In fact, due to the small size of each client's local data in a federated learning scenario, a stochastic gradient practically represents the full gradient derived from a client's data. This breaks the assumption on the identical gradient's distribution and makes it infeasible to carry out the data shuffling and aggregation processes of clients' gradients, which are used to avoid local minima [80]. Four main techniques are used to address the distribution imbalance challenge, i.e., *multi-task learning*, *client clustering*, *transfer learning* and *parameter tuning*. In the following, we discuss each of these techniques in detail and shed light on the approaches that employ these techniques.

Multi-Task Learning: In traditional machine learning, the objective is to solve an optimization problem for one particular metric. To do so, a single model (or sometimes an ensemble of models) is trained and fine-tuned until reaching a stable performance. While this approach has proved to yield good performance in a variety of domains, staying limited to only one task at a time might also deprive us from valuable information that could better optimize the original metric. In fact, by training several related tasks and sharing representations among them, the machine learning model can better generalize on its original task. This approach is referred to as Multi-Task Learning (MTL). For example, in the process of training students to be researchers, in addition to teaching them scientific research methodologies and techniques, we teach them writing, communication and presentation skills. Although these appear to be unrelated tasks, they turn out to equip the researcher with invaluable skills that are relevant to the original task, i.e., learning research. MTL can be practically implemented using two main methods, i.e., hard parameter sharing and soft parameter sharing. In hard parameter sharing which is mainly used to reduce overfitting, the hidden layers are shared among all tasks and several task-specific output layers are kept (Figure 9a). On the other hand, in soft parameter sharing, each task has its own model that keeps its own parameters. The distance between the different models' parameters is then regularized (e.g., using ℓ_2 norm regularization) to push them to converge (Figure 9b). In federated learning, soft parameter sharing MTL has been widely used to deal with the non-IID data challenge through assigning a different task model to different clients. Based on the assumption that some sort of structure considerably exists among different clients' task models (e.g., users may have similar behavior when using their mobile phones), the main idea of federated MTL is to capitalize on the relatedness (e.g., graph-based relatedness, sparsity, etc.) among the different tasks in order to capture the relationships among the non-IID data.

For example, the authors of [60] propose a multi-task learning-based approach that enables each client to learn a separate model,

in an attempt to fit separate but related models simultaneously. This approach is complemented by a resource-aware optimization model called *MOCHA* that takes into account the storage, computational, communication and power capacities of each client. As stated by the authors, *MOCHA* can only be applied to convex deep learning models and cannot be applied to non-convex models. In [61], the authors improve upon *MOCHA* [60] through incorporating non-linear mappings into the design of multi-task federated learning. This objective is accomplished through the inclusion of kernels to capture non-linear relationships in local training models as well as in the relationships between them.

Client Clustering: The client clustering approach aims to group clients into clusters that share common characteristics. The ultimate goal of such an approach is to facilitate the adoption of MTL in the federated learning framework. Specifically, the objective of federated MTL is to assign each client with a model that optimally fits its local data distribution. This, however, contradicts with the federated learning vision wherein all clients are treated equally and only one single global model is learned. To bridge this gap, the objective of client clustering approaches is to group clients into disjoint groups and hence assign a single task model to each cluster. The clustering is achieved through assessing the similarity among the clients' gradient updates. The main intuition is that clients with similar gradients tend to have similar data distributions. Consequently, clients that belong to the same cluster can jointly train a single model, which can be considered one task of the whole federated MTL.

For example, the authors of [62] depart from the observation that federated learning yields suboptimal results in case where the data distributions of the clients' local data are divergent to propose a clustering strategy that takes advantage of geometric properties of the federated learning loss surface to cluster the clients into groups with jointly trainable data distributions. The idea is that the cosine similarity among the weight updates of the different clients is a hint about the similarity in their data distributions. In [63], the authors propose a statistical model that takes into consideration the cluster structure of clients' data. The main component of this model is a Lloyd clustering algorithm that employs $_2$-distance to group clients' data in the presence of adversarial data observations. Based on the clustering algorithm, a heterogeneous federated learning optimization problem is designed and solved. The solution indicates that the proposed model matches the lower bound on the estimation error in terms of both dimension and number of data observations. In [64], the authors propose to group the clients based on the similarity of their local updates to the global joint model using a hierarchical clustering mechanism. The clusters are independently trained in a parallel fashion on specialized machine learning models.

Transfer Learning: Transfer learning is a machine learning approach that, instead of initiating the training process from scratch, gives the chance to start from patterns that were already learned through solving a different problem. For example, if you trained a simple classifier to predict whether an image contains a table, you could use the knowledge that the model gained during its training to recognize couches. Transfer learning addresses the problem of performance degradation in supervised learning when we modify the domain or task but do not have enough labeled data for that new domain or task. For instance, suppose that we trained a model to detect animals on images taken during the daytime. In this scenario, the task is detecting animals and the domain is that of daytime. Now, if we want to train another model to detect animals in images taken during the night-time (a different domain), we would need millions of labeled images taken during the night-time. The performance of the model trained on daytime images would considerably degrade when applied to night-time images since this model does not know how to generalize to a new domain, i.e., night-time.

In general, the transfer learning vision is achieved through employing a pre-trained model, i.e., a model that has been trained on a large dataset to solve a problem that is similar to the one you want to solve. We first train a base source network on a base dataset and a base task. Then we re-purpose the learned features (or transfer them) to a second target network to be trained on a target dataset and task. Basically, we transfer the weights that the model has learned from "task A" to a new "task B". This process is beneficial provided that the features are general, i.e., appropriate for both the base and target tasks, rather than being specific to the base task. Technically speaking, transfer learning benefits from the fact that the features in the lower levels of the deep network are highly transferable. This is because these features focus on learning common and low-level features. Hence, the parameters that are obtained from the source model can be transferred to the target model so as to learn the personalized features of that model. The general idea of transfer learning is schematized in Fig. 10.

Transfer learning is used in the context of federated learning to address the limited applicability problem of federated learning to only vertically or horizontally partitioned data. Given that most existing datasets have a small number of common features and/or population and a majority of non-overlapping data [81], [82], most of the data would be under-utilized by the federated learning framework. Transfer learning is proposed as a potential solution to such a challenge as this approach does not impose any limitation on the distribution of the data and can hence provide results for the entire population and feature space. For example, the authors of [65] propose a privacy-preserving federated transfer learning approach to deal with the heterogeneity in the clients' data distributions. In this approach, a model is learned from a source-domain's data distribution to be applied on a different (yet related) target-domain's data distribution with high accuracy. The authors complement the federated transfer learning approach with a secure transfer cross-validation model which employs additively Homomorphic Encryption (HE) to protect the performance of the federated transfer learning approach from the security perspective. In [66], the authors extend and improve upon [65] in two ways (1) reducing the overhead of the security model by an order of magnitude through employing secret sharing instead of homomorphic encryption and (2) extending the semi-honest secure multi-party computation model to consider dishonest malicious parties that might arbitrarily deviate from the federated training process. The SPDZ secure multi-party computation protocol is investigated in this context.

Parameter Tuning: Parameter tuning approaches generally seek to improve the convergence of the federated training on non-IID data through modifying the stochastic gradient descent method to improve its global convergence time. Two main strategies are adopted for this purpose. The first one is that of the momentum and the second is that of parameter correction. The main objective of the momentum strategy is to push the gradient vectors in the right directions. Momentum is a method of weighed averaging whose main idea is to *denoise* a series of data through moving the average of this series to bring it closer to the original function (e.g., exponential, linear, etc.). In federated learning, SGD with momentum consists of using the momentum method to average over the gradients received from clients at different iterations in order to help push the gradient vectors to go in the right direction. For example, the authors of [67] propose Momentum Federated Learning (MFL), a model that capitalizes on the momentum term in the local update phase. Instead of relying on the traditional gradient descent method in which the next iteration takes into consideration only the current gradient, MFL considers the last iteration along with the current gradient to accelerate the convergence. At the end of each iteration, the server performs a weighted averaging over the received local parameters to derive both the global momentum parameter and global model parameter and send them back to the clients. In [68], the authors discuss a consensus-based distributed momentum SGD model that satisfies decentralized computation, data parallelization

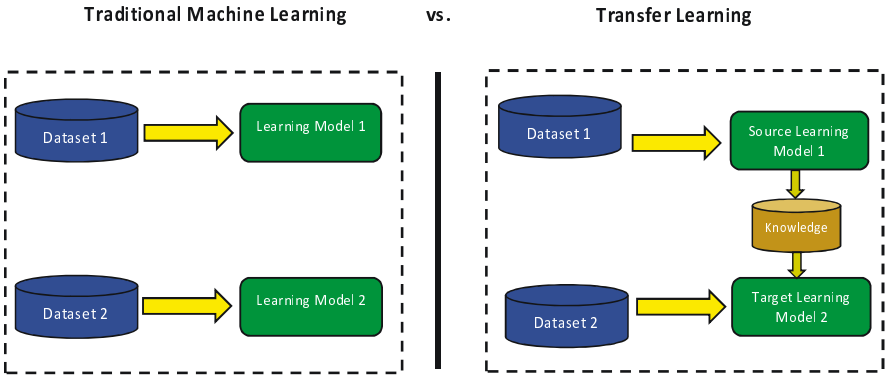


Fig. 10: Transfer Learning: The machine learning model capitalizes on some patterns that were already learned through solving a previous problem.

and constrained communication. The convergence of the proposed model is analyzed for both fixed and diminishing step sizes using a Lyapunov function construction approach, showing that the proposed model gives a convergence rate that is similar to a centralized model, while on the other hand boosting data parallelism and distributed computation.

The second strategy under the parameter tuning technique is that of parameter correction and, as you might have guessed, it consists of "correcting" some of the parameters to achieve some specific goals. In [69], the parameter subject to correction is the learning rate and the objective is to try to push the global model to be optimal for each client's local data at the beginning of each training iteration. In [21], the local subproblem of each client is subject to correction and the objective is to restrict the impact of local updates and force them to be closer to the initial global model. In [70], the parameter subject to correction is each client's local gradient and the objective is to improve the gradient aggregation process through propelling the median of gradients to draw near the expected mean. In [24], the parameter subject to correction is the neurons of the neural network and the objective is to enable global model size adaptation. In [71], the hyperparameters are subject to correction and the objective is to reduce the divergence of the local models. We explain hereafter each of these approaches in more detail.

In [69], Koskela et al. address the difficulty of carrying out validation in federated learning. The difficulty stems from the heterogeneity of the clients' data, which makes it difficult to generalize the hyperparameters across clients. To address this problem, a learning rate adaptation mechanism is proposed. In this mechanism, the learning rate is automatically computed using the moments accountant technique. The main motivation for adapting the learning rate is that the global model (after averaging over the clients' local models) may be far from optimum when it comes to clients' local data, which demands some adaptations at the beginning of each iteration. The authors of [21] propose *FedProx*, a variation of *FedAvg*, to tackle the statistical heterogeneity that stems from the non-identical distribution of data across devices. To address this problem, *FedProx* adds a proximal term to the local subproblem that is to be solved on each client's device. The objective is to restrict the influence of local updates through forcing them to be closer to the initial global model, while tolerating variable amounts of local work on the devices. In [70], the authors first investigate the convergence of *signSGD* and *medianSGD* in federated learning environments under clients' heterogeneous data distributions. The main idea of *signSGD* is to update the parameters according to a majority voting scheme on the sign of the gradients. On the other hand, *medianSGD* improves the robustness of *signSGD* through employing a coordinate-wise median of the gradients to estimate the mean of these gradients. The authors first demonstrate that both methods do not converge in heterogeneous data settings and then propose a gradient correction approach whose main idea is to perturb the local gradients with unimodal and symmetric noise, thus pushing the expected median to draw near the expected mean of gradients. The authors of [24] address the performance degradation of the averaged global model in FedAvg that arises from the coordinate-wise averaging of weights. The problem stems from the permutation invariance of the neural network parameters (i.e., many variants that differ only in terms of parameter ordering exist for any neural network). To address this problem, the authors propose Federated Matched Averaging (FedMA), a layer-wise federated learning model that considers the permutation invariance of the neurons prior to aggregation, thus allowing for global model size adaptation. The authors of [71] propose two strategies to address the data heterogeneity problem, i.e., (1) adaptive online hyperparameter tuning and (2) punishment for divergent representation learning across clients using a regularization term. The hyperparameter selection problem is formulated as an online reinforcement learning problem where, at each round, learners adopt an action through selecting certain hyperparameter values and then at the end of the round, each learner receives a reward which quantifies the reduction in training loss.

3.1.3 Size Imbalance

This type of imbalance is common in federated learning since clients are expected to have varying sizes of training data. Depending on the geographical location, gender, frequency and duration of activity and other factors, some clients might have considerably larger amounts of training data than other. For example, in an image recognition scenario, mobile phones owned by female users tend to have a larger number of photos compared to mobile phones owned by male users. Two main techniques are used to address the size imbalance challenge, i.e., data augmentation and flexible client participation. In the following, we discuss each of these techniques in detail and shed light on the approaches that employ these techniques.

Data Augmentation: Data augmentation techniques are used to augment the local data of the clients that considerably have lower volumes of data than others, to make the overall data more balanced in terms of size. Nonetheless, the data augmentation cannot be done in an arbitrary fashion as arbitrarily augmenting data might result in increasing the class imbalance across clients' data in case the newly added data are skewed towards certain classes. Theretofore, a data augmentation technique that aims to handle size imbalance should also be tailored to deal with class imbalance to avoid adding unnecessary or skewed data that might aggravate the class imbalance problem. Thus, the data augmentation approaches [55], [56], [57], [75] that are used in the literature to address the class imbalance problem, which

are described in the context of the *class imbalance* challenge (Section 3.1.1), are also used to deal with size imbalance.

Flexible Client Participation: The main idea of this approach is to relax some of the restrictions and constraints on clients' participation to make it easier for more clients to join the federated training. The objective is to attract a larger number of clients having data that match those that are held by clients having less volumes of data than the rest. For example, in [72], the authors propose to relax some of the restrictions on devices' participation through integrating the following four scenarios: (1) in-completeness, where devices are allowed to deliver partially completed work in a certain iteration; (2) in-activeness, where devices are allowed not to reply to the server in a certain iteration; (3) early departures, where devices can leave the training without completing all the training iterations; and (4) late arrivals, where new devices are allowed to enter after the training has already begun. In [73], the authors propose an experience-based federated learning framework called *FAVOR* whose main purpose is to counterbalance the effect of imbalanced data. The proposed framework intelligently selects the devices that are invited to contribute in each federated training iteration. To do so, a deep Q-learning [83] mechanism is proposed to thoughtfully choose, in each iteration, the subset of devices that maximizes the reward in terms of size balance and penalizes having more communication rounds.

3.2 Block Cycles

In the SGD method, it is essential for the data samples to be randomly drawn to preserve the performance. Cycling is a situation in which a specific permutation of data points occurs in an insufficiently-random order. An example of this situation includes highly correlated consecutive frames in a video. Although this situation can occur in centralized datasets that are hosted on a single device, it becomes even more complicated in distributed federated learning environments. This is because, in federated learning, only those client devices that are charging, idle and on a free wireless connection are chosen by the server to participate in the training. This creates considerable diurnal variation in the identity of available devices as these devices tend to meet the aforementioned participation eligibility criteria at night local time. This increases the possibility of encountering cyclic patterns in the data, where, for instance, the devices owned by French speakers in France and those owned by French speakers in Canada are likely to be available at different times of the day. The common approach to fight against block cycles in federated learning is that of *plurality*, which we discuss hereafter.

Plurality: Federated learning (as well as ensemble learning) solutions can be based on either a *consensus approach* or a *pluralistic approach*. A consensus approach strives to derive a consensus decision that is good for all the involved parties and, more specifically, it aims at creating one single predictor out of all the local models. On the other hand, a pluralistic approach embraces heterogeneity and allows to learn a separate model for each group of participants. Such an approach is useful in data that follows a block-cyclic structure, where a separate model can learned for each separate block of data. This, for example, can be achieved through linking each client with the hour at which the client is available for training. Then, instead of averaging over all iterations to get a single final predictor as is the case in consensus-based solutions, a collection of different pluralistic models can be created by averaging, for each data distribution associated with one particular block, over only the iterations that pertain to that block. Inspired by the idea, the authors of [74] first show that the presence of block-cyclic data largely deteriorates the training performance in federated learning. They then propose a pluralistic solution whose main idea is to train a different model for each block in the cycle, e.g., one model for daytime data and one model for nighttime data. Despite the importance of this solution, its efficiency is limited to convex machine learning optimization models and traditional SGD algorithms. Further investigations are needed to address this challenge in non-convex models and under parallel SGDs, which are the most frequently used in federated learning. In fact, most federated learning applications employ deep neural networks which rely on non-convex optimization models. Moreover, federated learning relies on a parallel SGD scheme (e.g., federated averaging) wherein the local gradients are aggregated in a parallel fashion.

3.3 Model Heterogeneity

Model heterogeneity refers to a situation in which each client independently designs its own training model. Clients in such a situation may not be willing to share details of their models due to privacy and intellectual property concerns. This scenario complies with small-scale federated learning environments (e.g., supply chain, healthcare) wherein a small number of participants want to design each a separate model to fulfill distinct unique specifications. The challenge here is to adapt the federated learning framework to a scenario where each client trains a different model that is a blackbox to the rest of clients. The concept of *knowledge distillation* is used to address this challenge.

Knowledge Distillation: Despite the abundance of complex deep learning models that can solve sophisticated tasks, a crucial challenge remains how to deploy such enormous models on small devices such as mobile phones for instant use. Knowledge distillation handles this challenge and aims to improve the performance of deep learning models on small devices. In other words, it aims at answering the following question: "How can we train a small network that can operate on resource-constrained devices?". The main idea of knowledge distillation is to train a large complex deep network that can elicit important features from the data and make accurate predictions. Thereafter, a small network is trained with the help of the large model so that this small network can replicate the results of the large model or at least produce similar results. Based on the hierarchical abstractions of features in deep networks, the small distilled model is trained to imitate the output of the large model instead of being directly trained on the raw data. From a conceptual point of view, the complex model is seen as a *Teacher Model* whereas the small network is considered to be a *Student Model*. Technically speaking, the transfer of generalization ability from the *Teacher Model* to the *Student Model* is achieved using the "soft targets" produced by the *Teacher Model*. Soft targets are the class probabilities produced by the output of the final softmax function from the source training model, whereas hard targets are the final output of the classifier. The soft target provides richer information for model training and can be used to restore intra-class variance and inter-class distance. In this way, the small model can be trained on a considerably less amount of data than that employed by the large model, while using a quite higher learning rate. While knowledge distillation may look similar to the concept of transfer learning explained earlier, they have two different purposes. In transfer learning, the weights are transferred from a pre-trained network to a new network whose architecture should exactly match the pre-trained network's architecture. This means that the new network should be as deep and sophisticated as the pre-trained one. On the other hand, in knowledge distillation, instead of transferring weights, we transfer the generalizations of the large model to a much smaller model. The general idea of knowledge distillation is illustrated in Fig. 11.

The authors of [75] employ knowledge distillation to address the parameter sharing challenge in heterogeneous local models environments. First, a data augmentation technique is employed to increase the volume of private data samples for each participant. This is done through allowing each participant to fully train its model using public data, prior to training it on its own private data. Thereafter, knowledge distillation is used to communicate Blackbox models

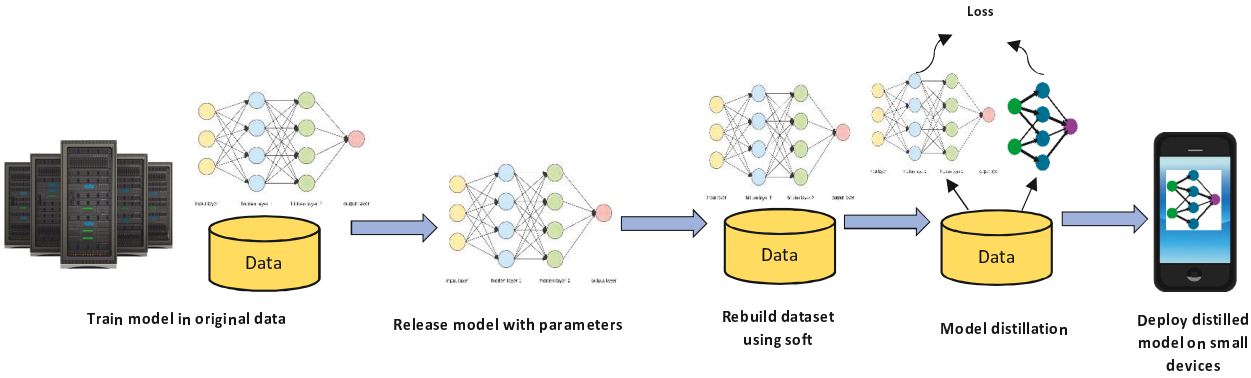


Fig. 11: Knowledge Distillation: A large complex deep network is first trained and then a small network is trained with the help of that large model to replicate its results on resource-constrained devices.

among participants without sharing neither private data nor model updates. This is achieved through letting each participant to share its knowledge in terms of class scores, derived from the public dataset. Then, the server gathers the class scores from all participants and computes their average. Based on this average, each participant trains its model again to approach the consensus (i.e., to make its class scores closer to the average).

3.4 Bias Mitigation

Bias recently began to attract a great attention in modern machine learning systems [84], [85]. This problem arises from a machine learning system whose outcomes (e.g., predictions) discriminate against some protected groups or visible minorities. For example, a linear regression model could base its salary predictions on the person's gender or ethnic group, leading to predict a lower salary for people from a certain group (e.g., females, people of color, etc.). Although many approaches have been lately proposed to fight against bias in traditional machine learning systems [86], [87], this topic has received less attention in the context of federated learning. Yet, bias is particularly challenging in federated learning since the target distribution based on which the global shared model is learned is unspecified. Adding to that the fact that the client participation in the federated training depends on many factors such as the device being charging, idle and on a free wireless connection. These factors combined may lead to a situation wherein the data used for training might not reflect the real data post-training. Let's better illustrate the idea with a real-world example. Suppose a federated learning scenario wherein the learner has access to a wide population of expensive mobile phones (80%) owned by technical users, and smaller population of general non-technical users (20%). In this case, the federated learning model would be mainly trained based on the uniform distribution of expensive devices, making it not representative of the general target domain where most of phones are owned by general non-technical users (e.g., 80% general versus 20% technical). The weighted optimization technique, explained in what follows, is used to fight against bias in federated learning.

Weighted Optimization: This method consists of assigning weights to the local models received by the clients. The weighting scheme can be done based on several criteria, e.g., giving more weights to devices with poor performance. The authors of [76] address the bias that arises from the fact that the central learning model is trained to minimize the loss function with respect to the uniform distribution over the union of all samples. The authors put forward an agnostic federated learning approach in which the shared learning model is optimized for any target distribution consisting of a mixture of client distributions. To solve the optimization problem, a fast-stochastic algorithm whose convergence bounds are proved is further proposed. In [77], the authors address the problem of fairness among edge devices model's accuracy in heterogenous federated learning environments. To this end, they propose a novel optimization problem called q-Fair Federated Learning (*q-FFL*), which boosts the fairness of the accuracy distribution across devices through assigning higher weights to the devices that generate higher loss. The idea is to boost less variance in the accuracy distribution. The *q-FFL* is solved using *q-FedAvg*, whose main idea is to employ a dynamic (instead of static) step-size stochastic gradient descent.

3.5 Desirable Criteria for Future Solutions

Based on the above classification and discussions, we identify a set of criteria that we believe are important to consider when designing future statistical solutions for federated learning. In what follows, we first present these criteria and then discuss how could they be practically capitalized on to craft efficient statistical solutions in federated learning.

- **Criterion** #1: Consider not only the first-order gradient but also preceding iterations to the current gradient update to accelerate the global convergence.
- **Criterion** #2: Support non-convex optimization models to avoid local optima in the presence of a large number of clients.
- **Criterion** #3: Handle block cycles in the training data to preserve the performance of the SGD method (Approaches [74]).
- **Criterion** #4: Decouple the local training from the global model aggregation to support local learners that use different machine learning algorithms.
- **Criterion** #5: Capitalize on data augmentation strategies to minimize the class, distribution and size imbalances (Approaches [55], [56], [57], [58]).
- **Criterion** #6: Design a weighted global aggregation model that assigns higher weights to learners that allocate higher computation capacities to the training process.
- **Criterion** #7: Differentiate between clients according to their training utility.
- **Criterion** #8: Capitalize on the similarity among the weight updates of the clients to detect the degree of similarity of their data distributions.
- **Criterion** #9: Investigate the impact of system metrics on the statistical heterogeneity.
- **Criterion** #10: Take into consideration the model heterogeneity in which each client trains a different machine learning model.

Criterion #11: Ensure that the federated training model is not biased toward or against any client's data distribution or device characteristics.
Criterion #12: Capitalize on the strength of transfer learning to address the limited applicability problem of federated learning to only vertically or horizontally partitioned data.

The current literature on statistical concerns in federated learning can be improved in many respects. To begin with, further methods to accelerate the convergence of the federated training are needed. The momentum method adopted by many approaches [67], [68] offers a great potential toward this objective. However, further investigations are needed on the interdependencies between this method and the other aspects of federated learning such as communication cost, security and privacy guarantees. A second aspect of research is to extend the existing literature that applies only on convex optimization machine learning models to cover non-convex models as well. In fact, despite the importance of the existing approaches that tackle statistical concerns in federated learning, most of them are dedicated to convex optimization models. Yet, most modern deep learning techniques (e.g., deep neural networks), which are the backbone of the federated learning paradigm, are based on non-convex optimization models. Therefore, it is of prime importance to adapt the existing solutions to non-convex scenarios to boost their applicability in modern machine learning settings. Another aspect to consider in the future is investigating the impact of system metrics such as amount of available resources on the non-IID degree of the data. Such information can be capitalized on to design more intelligent model aggregation techniques. For example, designing a weighted global aggregation model that assigns higher weights to clients that dedicate higher local computation resources to the training process can be quite beneficial to reduce both the non-IID degree of the data and straggler nodes that take long times to send their updates. Decoupling the local training from global model aggregation is another important aspect that needs careful investigations in the future. This is essential to dissociate the global aggregation process from the machine learning technique locally used by the clients and also to support situations wherein each group of clients prefers to use a different machine learning model. An additional research direction would be to design a training utility for each client. Such utility could be designed while taking into consideration a variety of metrics such as data size on the client's device, local computation time history and impact of the client's data on the non-IID degree of the overall data distribution. Then, based on this utility, the server can eliminate some clients from its selections to reduce the communication and computation overheads. Another essential research direction to continue to work on is that of detecting and alleviating the implications of the non-IID nature of the data. To do so, two interesting approaches need further analyses and investigations. The first approach is that of data augmentation, where thoughtfully created artificial data could be added to the clients' local data to mitigate the non-IID degree of the overall distribution. The second approach consists of analyzing the similarity and correlation between the clients' model updates to detect the similarity among their data distributions. Another research direction would be to detect and handle the problem of block cycles which mainly arises due to the system heterogeneity and changing client participation rate. Only one solution [74] has been proposed so far to address this problem. Despite the importance of this approach, its applicability is limited to only convex machine learning optimization models and traditional SGD algorithms. Therefore, further investigations are needed to adapt this approach to the federated learning settings, i.e., non-convex optimization models and parallel SGD computation.

4 Communication Efficiency (Challenge 2)

Communication efficiency approaches seek to reduce the communication overhead that arises from the exchange of messages between the parameter server and the client devices that run the model training in a distributed fashion. Two types of communications can be distinguished in this context, i.e., downstream communication and upstream communication [106], [107], [108]. Downstream communication refers to the process of clients downloading the current training model from the server, while upstream communication refers to the process of clients uploading the updated model to the server. Technically speaking, as discussed in [89], the number of bits that each client needs to upload and download during the federated training process is given in Equation (3):

$$bit^{up\ down} \quad O\ N_{iter} \qquad M \quad H \quad M^{up\ down} \tag{3}$$

where N_{iter} is the total number of local training iterations carried out by each client, is the communication frequency, M is the size of the training model M, $H \quad M^{up\ down}$ is the entropy of the weight updates exchanged during the upload and download operations respectively, and is the difference between the true update size and the minimal update size (given by the entropy). Thus, N_{iter} in Equation (3) represents the number of updates performed by each client and $M \quad H \quad M^{up\ down}$ quantifies the size of the update to be uploaded/downloaded by the client. Based on this formula, we can conclude that the communication overhead can be reduced in three ways, i.e., (1) reducing the size of the weight updates $M \quad H \quad M^{up\ down}$; (2) reducing the communication frequency N_{iter} ; and (3) communication type which affects both the weight updates size and communication frequency. In the following, we examine each of these three categories, i.e., *model updates size reduction*, *communication frequency reduction*, and *communication type* in detail and discuss the approaches that were proposed in the literature under each category. The classification scheme of the statistical approaches is schematized in Fig. 12. Moreover, we provide in Table 7 a summary of the main approaches that tackle statistical challenges in federated learning and highlight the criteria (proposed in Section 4.5) that each underlying approach satisfies.

4.1 Model Updates Size Reduction

The approaches that fall under this category seek to reduce the size of the model updates that are exchanged between the server and the involved clients. The common approach for doing so is the use of compression models. We explain hereafter how compression has been used in the context of federated learning and discuss in detail the solutions that were proposed in the literature to apply compression.

Compression: Compression refers to the process of encoding information using a number of bits that is smaller than that of the original representation [109]. Compression models can be divided into two categories, i.e., lossy compression models and lossless compression models. A lossy compression model is a compression process wherein some data from the original target (being compressed) is lost [110]. This process is irreversible in the sense that it would no longer be possible to go back once the target has been converted using a lossy compression model. On the other hand, a lossless compression model is a model in which the size of the target is reduced through taking out some unnecessary data (e.g., metadata) without entailing any quality loss. All of the existing compression models proposed for federated learning employ lossy compression methods to reduce the size of the weight updates. Specifically, the *quantization* and *sparsification* compression methods have been heavily used toward this end. Nonetheless, most of the existing approaches are hybrid and prefer to combine techniques from both quantization and sparsification to achieve high-quality compression. We discuss in what follows

TABLE 7: Summary of the communication efficiency approaches

Approach	Challenge	Technique	Main Idea	Criteria
Reisizadeh et al. [22]	Model Updates Size Reduction	Compression (Quantization)	A quantized message-passing scheme in which clients apply a quantization operator on the difference between their own updated model and the most recent model received from the server.	Criterion #2
Agarwal et al. [88]	Model Updates Size Reduction	Compression (Quantization)	A quantization method whereby clients quantize their locally computed gradients and send an efficient representation of the quantized gradient to the parameter server.	None
Sattler et al. [89]	Model Updates Size Reduction	Compression (Sparsification)	A sparse ternary compression model that extends the top-k gradient sparsification compression method to support both upstream and downstream communications.	Criterion #1
Amiri et al. [90]	Model Updates Size Reduction	Compression (Hybrid)	Compress the local gradients to a finite number of bits and then sparsify them into a low-dimensional vector using the superposition property of the wireless MAC.	Criterion #2
Konevcny et al. [9]	Model Updates Size Reduction	Compression (Hybrid)	Proposing two types of model updates: (1) structured updates which force the model update to follow a pre-specified structure and (2) sketched updates in which the full model is learned and then compressed using a mix of subsampling, quantization and random rotation.	Criterion #2
Caldas et al. [91]	Model Updates Size Reduction	Compression (Hybrid)	Employ techniques from basis transform, subsampling and probabilistic quantization to achieve lossy compression on the training model while maintaining its quality.	Criterion #2
Yurochkin et al. [92]	Communication Frequency Reduction	Parameter Number Reduction	A probabilistic nonparametric approach which identifies the subsets of neurons in each local model that match with the neurons in other clients' local models.	Criterion #3
Zhu et al. [93]	Communication Frequency Reduction	Parameter Number Reduction	A sparse evolutionary training model that determines and encodes the connectivity between every two neighboring layers of the neural network, with the purpose of decreasing the number of model parameters that need to be shared with the server.	Criteria #2 and #3
Niu et al. [94]	Communication Frequency Reduction	Parameter Number Reduction	A federated submodel learning framework in which clients need to download and upload only some relevant parts of the full model.	Criteria #2 and #3
Caldas et al. [91]	Communication Frequency Reduction	Parameter Number Reduction	Enable client devices to perform the local training on smaller subsets of the global model through setting to zero a steady number of activations at each fully-connected layers, which leads the weight matrices to be multiplied by a reduced number of activations.	Criteria #2 and #3
Liu et al. [25]	Communication Frequency Reduction	Periodic Aggregation	A client-edge-cloud aggregation model which enables each edge server to aggregate its own clients' model updates, and then periodically send aggregated models to the cloud server for global aggregation across all edge servers.	Criteria #4
Jeong et al. [58]	Communication Frequency Reduction	Periodic Aggregation	An online federated distillation technique in which each local device obtains the mean model output of all other devices and then periodically computes the difference between its model output and the obtained mean model output the using a cross entropy method.	Criteria #2 and #4
Guha et al. [95]	Communication Frequency Reduction	Periodic Aggregation	A one-shot federating learning approach that lets client devices to train local models to completion and then uses ensemble learning techniques to aggregates models across clients, instead of computing incremental updates.	Criteria #4
Liu et al. [96]	Communication Frequency Reduction	Periodic Aggregation	A federated stochastic block coordinate descent approach wherein clients constantly perform their local model updates and only occasionally share the updates with the server is proposed.	Criteria #2 and #4
Kamp et al. [97]	Communication Frequency Reduction	Periodic Aggregation	A selective communication scheme only in cases where the training model averaged over local learners suffers from loss.	Criteria #2 and #4
Reisizadeh et al. [22]	Communication Frequency Reduction	Periodic Aggregation	Periodic averaging where clients perform several local updates and only periodically synchronize their updates with the server.	Criteria #2 and #4
So et al. [23]	Communication Frequency Reduction	Periodic Aggregation	A multi-group circular strategy that divides clients into several groups and then at each aggregation iteration, the clients residing in one group forward the aggregated model updates of all clients in the previous groups and the local model updates of the current group to the clients of the next group.	Criterion #4
Wang et al. [98]	Communication Frequency Reduction	Periodic Aggregation	An in-edge federated learning framework that capitalizes on deep reinforcement to design a cooperation scheme among edge devices to exchange the learning parameters.	Criterion #4
Feng et al. [99]	Communication Frequency Reduction	Periodic Aggregation	A collaborative relay network scheme wherein only relay nodes connect to the access point of the server to forward the model updates.	Criterion #4
Sun et al. [100]	Communication Frequency Reduction	Over-The-Air Computation	An analog transmission scheme in which the summation of the local gradients is carried out over-the-air, instead of being performed on the parameter server.	Criterion #2 and #7
Yang et al. [101]	Communication Frequency Reduction	Over-The-Air Computation	A beamforming strategy that directs a wireless signal toward a specific receiving device, rather than broadcasting it in all directions.	Criterion #2 and #7
Amiri et al. [102]	Communication Frequency Reduction	Over-The-Air Computation	A distributed SGD method whereby only one device is selected at each iteration to compute its gradient and the average gradient is computed over-the-air.	Criterion #2 and #7
Wu et al. [103]	Communication Type	Asynchronicity	A semi-asynchronous federated averaging protocol that allows stragglers to remain asynchronous with the server so as to benefit from their progress in later iterations.	Criterion #2 and #6
Chen et al. [104]	Communication Type	Asynchronicity	An asynchronous online federated learning solution that adds a relatedness parameter to the global loss minimization function to measure the distribution similarity among clients which continuously receive new data during training.	Criterion #2 and #6
Pinyoanuntapong et al. [105]	Routing Scheme	Model-Free Reinforcement Learning	A model-free reinforcement learning approach that seeks to minimize the convergence time of federated learning in multi-hop environments.	Criterion #2 and #9

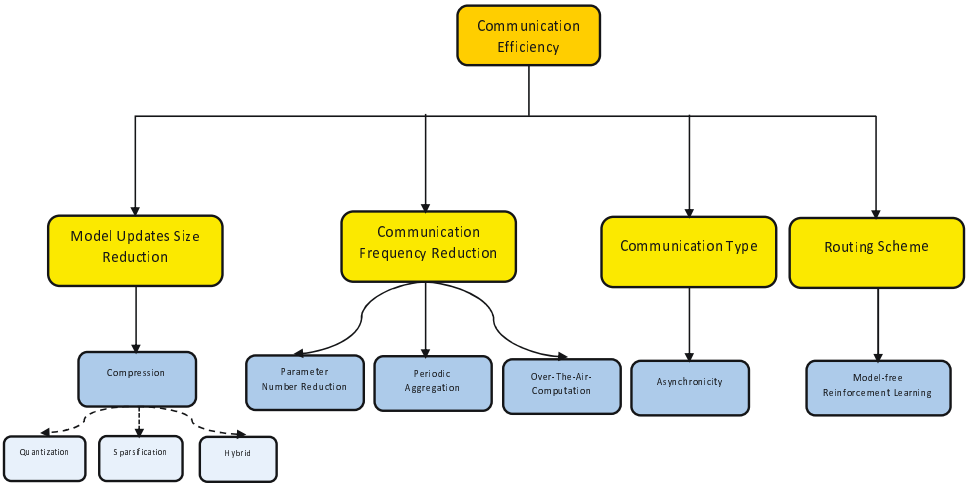


Fig. 12: Classification of the communication efficiency approaches in federated learning

the *quantization*, *sparsification* and *hybrid* techniques while shedding light on the associated approaches proposed in the literature.

Quantization: Quantization consists of applying a lossy compression on the gradient vectors through quantizing each of their entries to a finite-bit low precision value [111]. In general, quantization operates through compressing a set of values to a single quantum value. Thus, when the number of discrete symbols in a certain stream is decreased, the stream becomes more lightweight. As an example, decreasing the number of colors needed to represent a digital image would contribute in reducing its file size. The authors of [22] propose a quantized message-passing scheme in which clients apply a quantization operator on the difference between their own updated model and the most recent model received from the server and then submit then quantized vector to the server. The authors of [88] adopt a quantization method whereby clients quantize their locally computed gradients and send an efficient representation of the quantized gradient to the parameter server.

Sparsification: A sparse matrix is a matrix whose number of zero elements is higher than that of non-zero elements [112]. Sparsification is the process of making a matrix more sparse. The main advantage of having a sparse matrix is that, by using a matrix that is fundamentally composed by zero values, we can save space by storing only the non-zero elements [113]. In federated learning, sparsification is proposed to filter out the gradient vectors outputted by the clients according to their magnitude and select only a subset of them to be communicated with the server. For example, the authors of [89] propose a compression solution called Sparse Ternary Compression (STC) for both downstream and upstream communications. STC extends the top-k gradient sparsification compression technique to support both upstream and downstream communications using ternarization, sparsification, error accumulation and optimal Golomb encoding techniques.

Hybrid: Most compression-based solutions decide to embrace both the quantization and sparsification approaches and design hybrid compression models that involve both methods. The authors of [90] propose to compress the computed gradient to a finite number of bits and communicate these bits to the parameter server. Thereafter, they propose an analog communication scheme, called Compressed Analog - Distributed Stochastic Gradient Descent (CA-DSGD), that takes advantage of the superposition property of the wireless Multiple Access Channel (MAC) to enable devices to sparsify their gradient values and project these sparse values into a low-dimensional vector. The authors of [9] propose two types of model updates. The first is that of structured updates which forces the model update to follow a pre-specified structure. Two types of structures are discussed in this work, i.e., (1) low rank structure in which every local model update is forced to be a matrix of rank of at most k and (2) random mask in which the update is forced to be a sparse matrix that follows a predefined random sparsity pattern. The second type of model updates is that of sketched updates in which the full model is learned and then compressed using a mix of subsampling, quantization and random rotation, prior to transmitting it to the parameter server. The authors of [91] employ the techniques of basis transform, subsampling and probabilistic quantization to achieve lossy compression on the training model, while maintaining its quality.

4.2 Communication Frequency Reduction

The solutions that are proposed to reduce the communication frequency in federated learning can be classified into three major categories, i.e., parameter number reduction, periodic aggregation, and over-the-air computation. We discuss hereafter each of these categories in detail and explain the approaches that were proposed under each category.

Parameter Number Reduction: The commonly used approach for reducing the number of exchanged parameters is to tune the underlying deep neural network to decrease the connections among its layers. For example, the authors of [92] put forward a probabilistic nonparametric federated learning framework for neural networks. In this framework, the neurons of the clients' neural networks are matched prior to aggregation. The matching seeks to identify the subsets of neurons in each local model that match with the neurons in other clients' local models. The matching is achieved using the Beta-Bernoulli process (BBP) Bayesian nonparametric model, which allows local parameters to either be aggregated with existing global parameters or for new global parameters to get created in case the existing ones give poor matching. In [93], the authors seek to minimize the communication cost in federated learning without increasing the global model test error. This objective is modeled as a bi-objective optimization problem and solved using a multi-objective evolutionary algorithm. In more detail, a Sparse Evolutionary Training (SET) model is designed to determine and encode the connectivity between every two neighboring layers of the neural network. The purpose is to decrease the number of model parameters that need to be shared with the parameter server. The authors of [94] address the communication overhead caused by the fact that clients need to download and upload the full model learning at each federated learning iteration. To counter this problem, a federated submodel learning framework is devised, where clients need to download and upload only some relevant parts of the full model (referred to as submodels). In more detail, each client's submodel consists of the embedding parameters and parameters of the

network layers of the client's activities. In [91], the authors seek to enable client devices to perform the local training on smaller subsets of the global model. This is done through setting to zero a steady number of activations at each fully-connected layers, which implies that the weight matrices will be multiplied by a reduced number of activations.

Periodic Aggregation: This approach aims to reduce the number of times the clients need to communicate with the server through enabling these clients to perform multiple local iterations prior to sending their updated parameters to the server. For example, the authors of [25] investigate a client-edge-cloud federated learning aggregation model which enables partial model aggregation at the level of edge servers. The main idea is to enable each edge server to aggregate its own clients' model updates after every fixed number of clients' training iterations. Then, after every fixed number of edge model aggregation, the cloud server aggregates the models of all edge servers. Compared to a traditional cloud-based federated learning model, a client-edge-cloud model reduces the cost of communication with the cloud server and leads to decreasing the number of local iterations and overall runtime. In [58], an online federated distillation technique is discussed. According to this technique, each local device obtains the mean model output (a series of logit values normalized via a softmax function) of all other devices and periodically computes the difference between its model output and the obtained mean model output the using a cross entropy method. The authors of [95] leverage a one-shot federated learning approach which enables the server to learn a global model over a set of clients in one single communication iteration. The main intuition behind this approach is to let client devices to train local models to completion, instead of computing incremental updates. Consequently, ensemble learning models are implemented to seize global information across the device-specific models. The authors of [96] address the communication overhead in vertical federated learning that arises from applying privacy-preserving algorithms (e.g., Homomorphic Encryption, Multi-party Computation) on the communicated data. To attack this problem, a federated stochastic Block Coordinate Descent (BCD) approach wherein clients constantly perform their local model updates and only occasionally share the updates with the server is proposed. The authors of [97] propose to assess the need of communication based on the performance of the averaging model, aiming to decrease the communication overhead by an order of magnitude compared to periodic communication. In more detail, if the training model averaged over local learners suffers from loss, then communication is necessary; otherwise, communication should be avoided. The authors of [22] propose *FedPAQ* whose main idea is to perform periodic averaging, where instead of letting clients to synchronize their updated models with the parameter server at each iteration, these clients perform several local updates and then periodically synchronize their updates with the server. *FedPAQ* applies a partial device participation scheme as well, where only a subset of clients is selected to contribute in each training iteration based on several factors such as being idle, reachable to the base station, and connected to a free wireless network. In [23], the authors seek to reduce the overhead of applying the secure aggregation model in large-scale federated learning environments that may consist of billions of clients. To this end, a solution called *Turbo-Aggregate* is proposed. The first component consists of a multi-group circular strategy for model updates aggregation. The main idea here is to divide clients into several groups and then at each aggregation iteration, the clients residing in one group forward the aggregated model updates of all clients in the previous groups and the local model updates of the current group to the clients of the next group. The authors prove that such an aggregation strategy reduces the aggregation overhead from $O(N^2)$ to $O(N \log N)$. In [98], the authors design an in-edge federated learning framework that integrates together deep reinforcement, federated learning and mobile edge computing. The objective is to capitalize on the cooperation among edge devices to exchange the learning parameters, to both improve the training performance and reduce the unnecessary communications. The authors of [99] employ a relay network approach to reduce the number of clients communicating with the server. Mobile devices collaboratively provide relaying service to each other, where only relay nodes connect to the access point of the server to forward model updates. This leads in sub-centralizing the communications among the server and clients.

Over-The-Air Computation: Over-the-air [114] is an approach in which the whole or parts of the aggregation are carried out *over-the-air* by local devices to reduce the size of the messages communicated with the server. Traditionally, in federated learning, the summation of the gradients is carried out over-the-air in a distributed fashion among each group of clients and then only the sums (instead of all the gradients) are transmitted to the server to compute the average of the sums. For example, the authors of [100] propose an analog transmission scheme via multiple access channel, where the local gradient of each client is evenly partitioned into a set of segments. The summation of the local gradients is then carried out over-the-air (instead of being performed on the parameter server), under the assumption that all clients transmit their gradients synchronously. Moreover, an energy-aware client scheduling scheme is proposed with the objective of minimizing the average number of clients scheduled to submit gradient updates at each learning iteration under a long-term energy budget. The authors of [101] propose an over-the-air approach for efficient global parameters aggregation. The proposed technique takes advantage of the superposition property of a MAC to advocate a joint device selection model along with a beamforming strategy that directs a wireless signal toward a specific receiving device, rather than broadcasting it in all directions. These objectives are modeled as an intractable combinatorial optimization problem with non-convex quadratic constraints. The problem is solved using a Difference of Convex functions. The authors of [102] discuss a federated learning approach at the wireless network edge over a bandwidth-constrained fading Multiple Access Channel (MAC) path from limited power wireless devices to a remote parameter server. A distributed stochastic gradient descent technique is proposed, whereby only one device is selected at each iteration (based on channel conditions) to compute its gradient. The proposed solution takes advantage of the superposition property of the wireless channel to enable an over-the-air computation of the average gradient.

4.3 Communication Type

A communication type refers to the manner in which the server and clients communicate the training model and model updates. In federated learning, the communication between the server and clients can be either synchronous or asynchronous [115], [116]. In synchronous federated learning, the server needs to wait until all the local model updates from all clients are received prior to deriving an aggregate global model. On the other hand, in asynchronous federated learning, the aggregate model can be derived even if not all the local model updates have been received. The default communication type in federated learning is synchronous, since it complies well with the default aggregation model, i.e., *Federated Averaging*. The approaches proposed under this category attempt to craft an asynchronous communication model for the federated learning paradigm without violating this emerging paradigm's constraints. We discuss hereafter this technique which we refer to as *asynchronicity* and explain the approaches that are proposed under within its scope.

Asynchronicity: The approaches proposed under this category aim at overcoming the problems that arise from the synchronous communication model, which are the: (1) overhead caused by the need to distribute the latest global model over the entire set of clients;

(2) waste of progress in case some clients start the local training but fail to complete it; and (3) long waiting time in the presence of *stragglers* that take long time to report their local updates. Therefore, *asynchronicity* approaches aim to overcome the above-mentioned limitations through relaxing some of the synchronous federated learning assumptions. For example, the authors of [103] propose a Semi-Asynchronous Federated Averaging (SAFA) protocol that extends the *FedAvg* model. The solution capitalizes on asynchronous machine learning to reduce the impact of clients that drop offline (due to some power outage or low battery level) or are late in submitting the updated models (i.e., stragglers). To do so, the progress of the clients that are late in submitting their model updates on time is conserved to be used in later iterations. Thereafter, a discriminative aggregation technique that takes advantage of a cache structure is proposed to evade a portion of the clients' update and discriminatively combine some local models into a global model based on some criteria to accelerate the federated learning process. The authors of [104] address two essential challenges caused by the synchronous setting of federated learning. The first challenge relates to the volatile nature of each local edge device's training data, which allows for additional data to continuously come during the training process. The second challenge stems from a poor communication bandwidth on edge devices that might slow down the synchronized federated learning and waste the resources on the clients that need to wait until all other clients submit their model updates. To address these challenges, an asynchronous online federated learning solution is proposed, where updates from different clients with continuously arriving data can be combined in an asynchronous fashion. Technically speaking, the main idea is to add a relatedness parameter to the global loss minimization function to measure, in an online fashion, the distribution similarity among clients which continuously receive new data during training. Finally, to control the inconsistencies that arise from the asynchronous training model, the authors propose to enforce a dynamic learning step size on each client to control how often each client has to offer updates to the global model. Thus, edge devices with lower volumes of data and enjoying a stable communication bandwidth will be assigned a smaller dynamic learning step and hence a higher rate of contribution to the global model update process.

4.4 Routing Scheme

Traditional federated learning solutions capitalize on single-hop cellular links to convey the local updates from client devices to edge routers, which in their turn, connect to the remote cloud server via high-speed Internet core seeking for global model aggregation. Such a single-hop wireless communication architecture has the advantage of enabling edge devices to readily reach the cloud servers that are co-located with the cellular base stations. However, the main downside of such an architecture stems from the considerable deployment and operational costs that are required to create and maintain this type of networks. A wireless multi-hop communication architecture provides a more affordable and less complicated alternative to single-hop routing. A wireless multi-hop network architecture consists of a mesh of interconnected wireless routers, which are wirelessly connected using a mesh-like backbone structure. Adopting such a wireless multi-hop architecture in federated learning can bring many advantages. First, it improves the machine learning experience for mobile users in urban areas that are crowded with tall buildings, through reducing the line-of-sight problem[14]. Second, it helps democratize the machine learning experience through making it accessible to a larger mass of individuals at a lower cost, even those that are located in under-developed and disastrous regions. In the literature, few approaches have tried so far to adapt the multi-hop routing scheme to the federated learning design using the concept of *model-free reinforcement learning* which we discuss hereafter.

Model-free Reinforcement Learning: A model-based reinforcement learning algorithm is an algorithm in which an agent attempts to understand the environment and generate a model to represent it. This is done using two functions, the transition function (from one state to another) and the reward function. Thus, the agent has a clear reference which it uses to take decisions. In contrary, in model-free reinforcement learning, the agent does not make use of the transition and reward functions, but instead it follows a process of trial-and-error. The most widespread example of model-free reinforcement learning is that of Q-learning [83]. In federated learning, model-free reinforcement learning is used to replace closed-form mathematical models whose application is impractical in multi-hop settings. In fact, the existing efforts on wireless optimization in federated learning systems consider only single-hop federated learning scenarios over cellular edge computing systems. To do so, closed-form mathematical models are formulated to study the impact of wireless communication control parameters such as transmission power on the performance of the federated learning solution (e.g., in terms of update transmission delay). But alas, these mathematical models cannot be directly applied to multi-hop federated learning settings, where the federated learning performance metrics (e.g., convergence time) can no longer be represented as a closed-form function of the communication control parameters. Yet, optimization models are crucial in this emerging multi-hop architecture of federated learning to study the impacts of multi-hop routing on the local training time and global convergence processes. In fact, most federated learning aggregation techniques (e.g., *FedAvg*, *FedProx*) are synchronous in the sense that the server must wait for and collect all (or at least a minimum number of) local updates from the client devices prior to executing the global aggregation and moving to the next iteration. In such a scenario, a long and random multi-hop delay can drastically augment the number of *stragglers*, leading to longer training time per iteration. Moreover, applying a multi-hop communication scheme in such a fat federated learning architecture brings the risk of making the routing paths toward the cloud server to be readily saturated, resulting in a slower convergence speed. In this context, the authors of [105] seek to minimize the convergence time needed to achieve the required training accuracy. To do so and due to the difficulty of applying closed-form functions in multi-hop environments, a model-free reinforcement learning approach is proposed. In this approach, the distributed routers capitalize on their instant local experiences to cooperatively and on-the-fly tune the networking parameters.

4.5 Desirable Criteria for Future Solutions

Based on the above classification and discussions, we identify a set of criteria that we believe are important to consider when designing future communication efficiency solutions for federated learning. In what follows, we first present these criteria and then discuss how can they be practically capitalized on to craft efficient statistical solutions in federated learning.

Criterion #1: Consider both upstream and downstream communications in the design of the solution.
Criterion #2: Reduce the size of the model updates without sacrificing the model's accuracy.
Criterion #3: Design efficient solutions that detect and decrease the connections among the deep network's layers to reduce the number of model updates that need to be shared with the server.
Criterion #4: Support periodic aggregation where clients are allowed to perform several iterations prior to uploading their updates to the server.

14. Line-of-sight is a term used to depict an object that prohibits a person's sight path of their target destination.

Criterion #5: Support dynamic aggregation where the aggregation at the level of the server is performed only when judged necessary.
Criterion #6: Be resilient to large numbers of clients and unstable client participation.
Criterion #7: Support over-the-air computation to enable partial aggregation (e.g., summation of local gradients) to be done locally.
Criterion #8: Investigate device-oriented event-triggered communication schemes to adapt clients' devices to the training workload.
Criterion #9: Investigate multi-hop routing schemes to help democratize the federated learning experience and enable federated learning process in crowded urban areas.

The current literature on communication efficiency in federated learning can be improved in many aspects. First, it is important to investigate dynamic aggregation schemes where the aggregation at the level of the server would be performed only when judged necessary. The adoption of such a scheme needs however a comprehensive and systematic analysis on the trade-off between accuracy and communication. Specifically, we need communication techniques that can prove improvements at the Pareto frontier, thus giving an accuracy that is larger than any other approach under the same communication budget and across a large domain of communication/accuracy profiles [13], [117]. The communication schemes should also be robust to both large and unstable client participation. In fact, the main premise of federated learning is to allow the largest possible number of clients to participate in the federated training to enrich the model with a variety of data and hence improve its generalizability in the testing phase. Moreover, the participation of the clients in the federated learning paradigm is not always guaranteed. As discussed earlier in the paper, this participation depends on several factors (e.g., the device being plugged in, connected to WiFi, etc.). Moreover, this participation varies from one training iteration to another. For example, the participation in the first iteration might reach 90% of the invited clients while in the second iteration it might go down to 45%. Therefore, the communication scheme should be resilient to this variation from one task to another and also from one training iteration to another. One more research direction is to think of more advanced communication models. In fact, different from traditional worker nodes that are deployed in data centers which are workload-dedicated, the client devices in federated learning are neither workload-dedicated nor workload-ready. In more detail, worker nodes in data centers are programmed to always be ready to take their next job from the central node directly after delivering the results of their current job. On the other hand, in federated learning scenarios, the devices are general-purpose devices on which the server has not control. Therefore, it would be interesting to investigate device-oriented communication schemes that enable each device to take the decision on when to get activated and start interacting with the server in an event-triggered fashion.

5 Client Selection and Scheduling (Challenge 3)

The objective of client selection and scheduling approaches is to enable federated learning model owner (i.e., parameter server) to make thoughtful decision as to which clients to select and how to distribute the training tasks among them in such a way to improve the performance of the collaborative training and minimize the convergence time. Four main sub-challenges are addressed in this context, i.e., (1) resource management; (2) client number maximization; (3) client dropout and (4) reliable client selection. In the following, we examine each of these challenges in detail, explain the techniques that are used to address each particular challenge, and discuss the approaches that were proposed in the literature under each technique. The classification scheme of the client selection and scheduling approaches is schematized in Fig. 13. Moreover, we provide in Table 8 a summary of the main approaches that tackle challenges related to client selection and scheduling in federated learning and highlight the criteria (proposed in Section 5.5) that each underlying approach satisfies.

5.1 Resource Management

A major challenge that plays a critical role in the client selection and scheduling process is how to efficiently administer the limited resources on client devices to maximize the federated training performance [128]. Specifically, the amount of resources available on the client devices is a key factor for deciding which clients to select and the amount of work that should be assigned to each single client. The challenge here stems from both the large size and growing number of model parameters and the system heterogeneity and bandwidth limitations on the client devices. In fact, in the current deep neural networks, every model update can be in the range of gigabytes [57] and millions of parameters might need to be exchanged at each federated learning iteration. Moreover, the computation capacity and wireless link quality vary across clients. This raises the importance of designing resource management approaches to derive appropriate decisions as to how minimize resource consumption on the clients while at the same time improving the training accuracy. Even though plenty of resource management approaches have been proposed in the contexts of fog, edge and mobile cloud computing, these approaches cannot be directly applied in federated learning because of this emerging paradigm's unique characteristics. Specifically, not only the computation and communication capacity and data size on the clients should be considered, but also the distribution of the data across clients. Thus, the current resource management approaches seek to reduce the resource consumption in terms of energy and bandwidth on the participating client devices without significantly scarifying the training quality and convergence time. Energy consumption is caused by the process of computing model updates on the client devices, while bandwidth consumption is caused by the processes of downloading the learning model and uploading the updated model parameters from/to the server. The existing resource management approaches formulate the resource utilization minimization challenge as an optimization problem and employ techniques from *heuristics* and *reinforcement learning* to solve it. In the following, we discuss each of these techniques in detail and explain the approaches that were proposed under each particular technique.

Heuristics: A heuristic approach is problem-solving approach which uses a practical method that is not guaranteed to be optimal, but which is adequate for attaining an immediate short-term goal [129]. Heuristic methods are widely used to accelerate the process of deriving a satisfactory solution in cases where finding an optimal solution is infeasible or impractical [130]. In federated learning, heuristic methods are used to solve NP-hard resource management problems formulated as optimization problems. For example, the authors of [118] investigate the implementation of federated learning algorithms over practical wireless networks while jointly taking into consideration parameters from both federated learning and wireless networks. The main contribution of this work is studying the impact of wireless network conditions on the performance of federated learning. The problem is formulated as an optimization model in which the central server (base station in this work) seeks to optimize its resource allocation (in terms of bandwidth uplink) and clients seek to optimize their transmit power allocation. This optimization problem is then simplified to a mixed-integer nonlinear programming problem through deriving a closed-form expression for the expected convergence rate of the federated learning algorithm. The simplified

TABLE 8: Summary of the client selection and scheduling approaches

Approach	Challenge	Technique	Main Idea	Criteria
Chen et al. [118]	Resource Management	Heuristics	An optimization model in which the central server seeks to optimize its bandwidth uplink and clients seek to optimize their transmit power allocation.	Criteria #1 and #2
Wang et al. [4]	Resource Management	Heuristics	A control algorithm to derive a balance between minimizing the loss function while at the same time respecting a given budget of resources on the edge nodes.	Criteria #2 and #4
Hu et al. [119]	Resource Management	Heuristics	A segmentation approach in which the learning model is split into subsets, each consisting of the same number of non-overlapping model parameters.	Criteria #2 and #5
Li et al. [21]	Resource Management	Heuristics	A resource mapping mechanism that links the amount of work to be carried out on each device with the amount of available resources on that device.	Criteria #2 and #4
Zhou et al. [120]	Resource Management	Heuristics	A cost-efficient federated learning framework to reduce the computation cost on resource-constrained edge nodes using a coordination scheme between the edge and cloud layers.	Criterion #2
Nguyen et al. [121]	Resource Management	Reinforcement Learning	A Double Deep Q-Network approach that helps the model owner derive optimal decisions in terms of (1) amounts of energy to be recharged for clients and (2) appropriate channels for the global model transmissions so as to maximize the number of successful transmissions while at the same time minimizing the energy and channel costs.	Criteria #2 and #6
Anh et al. [122]	Resource Management	Reinforcement Learning	A Deep Q-Learning approach that enables the server to derive optimal resource management decisions without a priori knowledge of the resource status on the client devices.	Criteria #2 and #4
Nishio et al. [123]	Client Number Maximization	Multi-Objective Optimization	A multi-objective optimization approach that aims to find a tradeoff between increasing the number of participating clients and the inherent waiting time for local model updates to arrive.	Criteria #11 and #12
So et al. [23]	Client Dropout	Redundancy	A redundancy mechanism that employs Lagrange coding to allow clients to reconstruct the aggregate model in case of dropout.	None
Wu et al. [103]	Client Dropout	Asynchronicity	A lag-tolerant distributed algorithm that allows some clients to remain asynchronous with the central server to benefit from stragglers in later iterations even if they can't submit their updates on time.	Criteria #4 and #8
Kang et al. [124]	Reliable Client Selection	Trust and Reputation	A reputation mechanism that relies on direct opinions from the concerned model owner and indirect opinions from other model owners to derive a reputation score for each client device.	Criteria #8 and #10
Kang et al. [125]	Reliable Client Selection	Trust and Reputation	A reputation mechanism that maps the computation time on each client device to the size of the data on that device to identify the lazy clients that give poor performance.	Criteria #8 and #10
He et al. [126]	Reliable Client Selection	Trust and Reputation	A server-free decentralized federated learning approach that considers unidirectional trust relationships among clients, which are represented as a social network.	Criteria #8 and #9
Wang et al. [127]	Reliable Client Selection	Trust and Reputation	A mechanism that employs training instance omission in the case of horizontal learning and Shapley value in the case of vertical learning to measure the contribution of each client.	Criterion #7

problem is converted again into a bipartite matching problem and accordingly solved using a Hungarian algorithm [131]. The authors of [4] tackle the problem of efficiently exploiting the limited resources available on the edge networks to maximize the learning efficiency in federated learning scenarios. They propose a control algorithm to derive the best balance between the local update and global aggregation processes of gradient-descent based federated learning algorithms. The objective is to minimize the loss function while at the same time respecting a given budget of resources on the edge nodes. In [119], the authors aim to improve the network bandwidth utilization in federated learning without sacrificing accuracy. To do so, a segmentation approach is introduced, where the learning model is split into subsets, each consisting of the same number of non-overlapping model parameters. Clients then perform a segmentation-level model update by aggregating their local segments with those of k other clients. The authors indicate that the value of k should be less than the number of all clients to achieve good convergence. Additionally, a gossip-based protocol is proposed, where each client stochastically picks out a number of clients to which the local model segment is to be transferred. The authors of [21] propose *FedProx*, a variation of *FedAvg*, to tackle the system heterogeneity, which arises from the variability of system characteristics on the devices running the federated training. To overcome this challenge, *FedProx* links the amount of work to be carried out on each device with the amount of available resources on that device. In [120], the authors aim to reduce the computation cost on resource-constrained edge nodes through proposing a coordination scheme between the edge and cloud layers to optimize the system-wide cost-efficiency. Technically speaking, a Cost-Efficient Federated Learning (CEFL) framework that leverages Lyapunov optimization theory is proposed to generate online near-optimal decisions in terms of data scheduling, admission control, accuracy tuning and load balancing for dynamically arriving training observations.

Reinforcement Learning: Reinforcement learning is a machine learning subfield which is interested in teaching a software agent on what actions to take in a certain environment so as to maximize the cumulative reward [132], [133]. It has been used in federated learning as an alternative to heuristics in solving optimization problems. The main advantage of reinforcement learning compared to heuristics is that the former includes a learning component which allows for continuous improvement in the quality of decisions over time. Moreover, the fact that the server does not have complete knowledge about the resource status on the client devices makes it hard for heuristic solution to derive effective decisions. Therefore, reinforcement learning is used as an alternative approach to model this uncertainty. For example, the authors of [121] tackle two resource

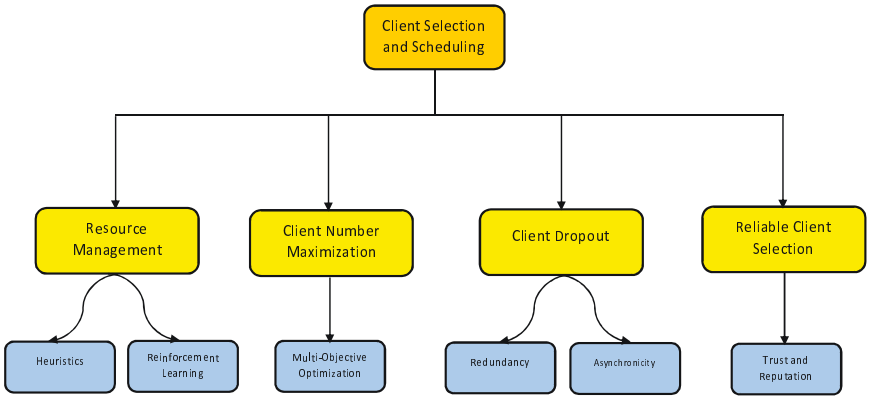


Fig. 13: Classification of the client selection and scheduling approaches in federated learning

management challenges in federated learning. The first challenge is related to the energy limitations on the mobile clients, which raises the problem of deciding on the appropriate amounts of energy that need to be allocated by the model owner to recharge the participating clients. The second challenge concerns the bandwidth cost entailed by the large number of global model transmission messages that needs to be exchanged among the clients and model owner. To answer these challenges, a *Deep Q-Network* approach [83] is proposed to help the model owner derive optimal decisions in terms of (1) amounts of energy to be recharged for clients and (2) appropriate channels for the global model transmissions so as to maximize the number of successful transmissions while at the same time minimizing the energy and channel costs. Technically speaking, the aforementioned problem is formulated as a stochastic optimization problem and then a Deep Q-Network algorithm with Double deep Q networks is designed to derive the optimal resource allocation policy. The resource limitation problem on client devices in terms of CPU, energy and bandwidth has been considered in [122]. The main challenge addressed in this work is the lack of complete knowledge about the resource status on the client devices, which prevents the server from selecting the appropriate clients to participate in the training process. This problem has been addressed using a Deep Q-Learning approach that enables the server to derive optimal resource management strategies without a priori knowledge of the resource status. This problem has been modeled as a stochastic optimization problem and solved using a Double Deep Q-Network technique.

5.2 Client Number Maximization

Intuitively, increasing the number of clients that participate in each training iteration contributes in reducing the time required for the global model to reach a desired performance. In fact, having more clients that participate in each round decreases the number of rounds that the parameter server has to launch to attain the required accuracy. Inspired by this idea, the objective of the approaches that fall under this category is to study the problem of maximizing the number of clients that participate in each training iteration in a thoughtful manner. This is however surrounded by several challenges that need to be taken into consideration such as the (1) impact of increasing the number of clients on the non-IID degree of the overall data; (2) impact of increasing the number of clients on the local model updates waiting delay; and (3) the reliability of the clients being added to the process. Although there is currently no approach that accounts for all these factors simultaneously, multi-objective optimization has been used in an attempt to simultaneously optimize some of these factors as will be discussed hereafter.

Multi-Objective Optimization: Multi-objective optimization is a decision-making approach that seeks to optimize more than one objective function for a given problem in a simultaneous fashion. It is often employed to derive trade-offs between two or more conflicting objectives [134]. An intuitive example of multi-objective optimization includes minimizing cost while maximizing quality in a car purchase scenario. In the client number maximization problem, multi-objective optimization has been used to optimize for more than one metric at the same time. For example, the approach proposed in [123] aims to derive a tradeoff between increasing the number of clients and the inherent waiting time for local model updates to arrive using multi-objective optimization. In more detail, the objective is to maximize the number of model updates received by as many clients as possible while improving the performance of the federated learning. The main idea is to fix a certain deadline for clients to download the parameters of a trainable model, update the model after training on their own data, and send the updated model parameters to the central server. Consequently, the server tries to maximize the number of clients to be selected to share their model updates within the specified time frame, while considering their computation and communication resource constraints.

5.3 Client Dropout

Client dropout refers to the problem of clients withdrawing from the federated training [135]. Dropout can occur at any moment of the training and can be due to several reasons such as low battery, poor connectivity, need to make phone calls, etc. A client that drops from the federated training near completion is referred to as a *straggler* and can cause a significant waiting time and resource wastage on both the server and rest of clients. The two common techniques to fight against client dropout are redundancy and asynchronicity. We explain these two techniques in detail and discuss the approaches that were proposed in the literature under each technique.

Redundancy: The main idea of this approach is to add redundancy to the model updates to be capitalized on for reconstructing the aggregated model in case some client dropout cases take place. Using this redundancy, clients in the later training stages can retrieve the information needed to cancel out the effect of dropped clients at the earlier stages without the need to carry out any extra communication. For example, the authors of [23] propose to add redundancy in the model updates using Lagrange coding [136]. The purpose is to fight against client dropout through exploiting the injected redundancy to reconstruct the aggregate model in the case of dropout.

Asynchronicity: The main idea of this approach is to embrace client dropout instead of fighting against it. This is done through relaxing the synchronicity assumption of the model aggregation process and hence allowing some clients to remain asynchronous with the server. In this way, stragglers can still contribute in future iterations of the federated training. The authors of [103] advocate a lag-tolerant

distributed algorithm that allows some clients to remain asynchronous with the central server. The objective is to benefit from stragglers in later iterations even if they can't submit their updates on time.

5.4 Reliable Client Selection

In federated learning, the training is divided over a large set of geographically distributed client devices. Since the model owner has no control over these devices, encountering unreliable clients in this machine learning paradigm becomes inevitable. Such clients would have catastrophic impacts on the federated learning process where, for example, unreliable clients might train their local models on unreliable data, leading to fraudulent learning models. Therefore, designing solutions to select reliable clients becomes crucial to guarantee the success of the federated training. Trust and reputation is the commonly used technique to address this challenge. We discuss hereafter this technique and highlight the approaches that rely on this technique to improve the selection of reliable clients.

Trust and Reputation: A trust and reputation model is a model that enables decision makers to distinguish good options from bad ones based on current interactions and historical experience [137], [138]. The importance of trust and reputation models in federated learning stems from its ability to enable the parameter server to differentiate between reliable clients and unreliable ones [139]. This helps the server make thoughtful selections so as avoid random choices in such an open environment which might expose the training process to quality, communication, and security problems. For example, the problem of selecting reliable mobile devices is tackled in [124]. First, a reputation mechanism is proposed to assess the reliability of the mobile devices. The reputation is computed using a subjective logic model wherein direct opinions from the underlying task publisher (central server) along with indirect opinions from other task publishers are combined to derive a reputation score for each device. The reputation management is achieved in a decentralized manner using the blockchain concept so as to ensure tamper-resistance and non-repudiation. The authors of [125] propose a reputation-based approach that relies on consortium blockchain to satisfy the non-repudiation and non-tampering properties is leveraged. In the first phase, federated learning model owners broadcast the tasks along with their specific requirements. Interested clients send joining requests to the owners along with relevant information about their identity and resources. In the second phase, the model owner computes a reputation score for each of these clients. The reputation is computed based on direct observations from past interactions as well as recommendations from other model owners stored on an open-access consortium blockchain. Finally, following the distributed training process, the model owner updates the reputation scores of the clients through mapping the computation time on each client device to the size of the data on that device to identify those "lazy" ones that gave poor performance. In [126], the authors propose a server-free decentralized federated learning approach based on an online Push-Sum algorithm. The proposed approach considers unidirectional trust relationships among clients which are represented as a social network (i.e., client X trusts client Y but client Y may not trust client X). As a first step, each client applies the current local model to derive the loss function, and subsequently computes an intermediate local model. Then, in the push step, a weighted variable of the intermediate model is forwarded to clients' out neighbors. Finally, in the sum step, all received weighted intermediate models are normalized and summed to derive the new local model. In [127], the authors design a mechanism to assess the contribution of each client in the federated training. The mechanism distinguishes between horizontal federated learning scenarios in which each client participates through providing a subset of the training instances and vertical federated learning wherein each client contributes a subset of the feature space. In the case of horizontal federated learning, the authors propose to omit the training instances from a particular client and then measure and compare the accuracy with and without these instances to get an idea of the actual amount of contribution of that client. In the case of vertical federated learning, the concept of Shapley value [140] from coalitional game theory is employed to compute the grouped feature importance, based on which the contribution of each client is then inferred.

5.5 Desirable Criteria for Future Solutions

Based on the above classification and discussions, we identify a set of criteria that we believe are important to consider when designing future client selection and scheduling solutions for federated learning. In what follows, we first present these criteria and then discuss how can they be practically capitalized on to craft efficient client selection and scheduling solutions in federated learning.

> **Criterion** #1: Study the impact of wireless network conditions on the accuracy of the federated training.
> **Criterion** #2: Derive optimal methods to efficiently utilize the limited computation and communication resource budget of client devices to maximize the training performance.
> **Criterion** #3: Learn and dynamically adapt the frequency of performing global model aggregation to minimize the resource consumption on the client devices.
> **Criterion** #4: Take into account the resource heterogeneity across client devices and make efficient use of this heterogeneity to optimize the training performance.
> **Criterion** #5: Allow for partial peer-to-peer model updates sharing, where clients synchronize the model updates with only a part of other workers without sacrificing the performance.
> **Criterion** #6: Account for the uncertainty and absence of a priori knowledge of the clients' resource status and network dynamics.
> **Criterion** #7: Evaluate the contribution of each single client to the federated training to both improve future selections and help designing efficient pricing schemes.
> **Criterion** #8: Assess the reliability of the participating clients to avoid undesirable behaviors and conditions.
> **Criterion** #9: Investigate the mutual trust relationships among clients to foster distributed local model updates sharing.
> **Criterion** #10: Protect the trust establishment process against malicious attackers that try to sway the trust results.
> **Criterion** #11: Investigate a tradeoff between increasing the number of participating clients and both the overall distribution of the data and resulting waiting delay for model updates.
> **Criterion** #12: Consider a variety of factors such as non-IID degree of data, data size, computational and network resource capacilities, and reliability in the client selection process.

The literature on client scheduling can be extended in many directions. The first direction would be to design optimal resource management solutions to efficiently utilize the limited computation and communication resources of the client devices in such a way to maximize the training performance. Despite the importance of the optimization solutions proposed in the literature to address this challenge, these approaches cannot model the interdependencies between the strategies of the server and those of the clients and hence cannot capture the whole picture of the problem. Yet, the strategies adopted by the clients (e.g., amount of resources dedicated to the local training) have significant influence on the profits (e.g., training accuracy) of the server and vice versa. Game theory offers plenty of models and techniques (e.g., maxmin, Stackelberg, etc.) [141], [142], [143] that can help overcome this limitation. The advantage of game theory over the traditional optimization techniques in such a scenario stems from

its ability to model this situation as a two-player game wherein one player (model owner) seeks to maximize the training accuracy and the other player (clients) seek to minimize the resource utilization. A second research direction would be to thoughtfully reduce the frequency of carrying out global model aggregation to decrease the resource utilization on the clients' devices. This can be done through learning and dynamically adapting this frequency based on the obtained and desired accuracy. One additional research direction is to embrace the resource heterogeneity across client devices to design intelligent resource management strategies that link the training load to be assigned to each client with the amounts of available resources on that client's device. This however necessitates designing solutions to motivate clients to truthfully reveal their actual resource capacities, where mechanism design (discussed later in the context of service pricing) offers interesting techniques toward this purpose. Another interesting research direction that needs to be taken into consideration is the uncertainty that the model owner faces in terms of absence of a priori knowledge of the current resource status of the clients' devices. This deprives the model owner from having exact and accurate information and hence negatively affects the precision of the resource management strategies. To tackle this problem, deep reinforcement learning has been lately investigated [121], [122] to model this uncertainty, where promising results have been obtained. Other approaches such as *Double and Duel Deep Q-Learning* and Bayesian game theory could also be used to improve upon these approaches. Another research direction to consider is the investigation of partial peer-to-peer model updates sharing, where clients synchronize the model updates with only a part of other workers instead of sharing them with the whole set of clients or repeatedly with the server. This can be accomplished using the concept of clustering from machine learning which can help uncover those groups of client devices that sharing the model updates among their members would have the best impact on the model's accuracy.

From the perspective of client selection, several aspects need to be considered in the future. To begin with, the literature needs approaches for deriving the optimal number of clients that need to participate in each federated learning iterations. Only one work [123] has addressed this challenge so far. Therefore, we need more comprehensive solutions that take into considerations several constraints of the federated learning paradigm in the design of the solution. These constraints include the impact of the number of clients on the non-IID degree of the overall data, the reliability of the clients being invited to join, and the impact of adding more clients on the waiting delay for model updates. Multi-objective optimization and game theory are good candidates to formulate this problem while considering these constraints in a simultaneous fashion. Another interesting research direction would be to propose novel methods to evaluate the contribution of each single client in the training process. This is important to both improve future selections and help designing efficient service pricing schemes. Last but not least, it is of prime importance to propose trust and reputation models to evaluate the reliability of the participating clients. Although some trust and reputation models already exist in this context, more comprehensive models are needed to cover a wider set of scenarios and constraints. In fact, it is important to consider the trust relationships not only between the server and clients but also among the clients themselves. Having such trust relationships would help design distributed model updates sharing models wherein clients share the model updates with each other for several iterations (e.g., periodic aggregation, over-the-air-computation, etc.) before sending them to the server to reduce the communication overhead. Moreover, it is also important to design solutions to protect the trust establishment process from malicious attacks that try to sway the trust decisions such as *camouflage*, *whitewashing*, *promoting* and *slandering* [144]. This is of prime importance to prevent unreliable clients from benefiting from bogus high trust scores to sneak into the system and launch security and privacy attacks against the federated training process or other clients' data.

6 Security Concerns (Challenge 4)

As a new machine learning approach, federated learning is target to a variety of attacks that aim to manipulate the collaborative learning process [153]. These attacks can be classified into two categories, i.e., targeted attacks and untargeted attacks. Untargeted attacks (often referred to as *Byzantine*[15]) aim to deteriorate the model's performance or to cause some failure in the training process in general without targeting any particular client or data samples. On the other hand, targeted attacks have clear malicious objectives and employ some sophisticated techniques to attain them. In the following, we shed light on each of these categories of attacks and discuss the existing techniques that are proposed to counter each of them. The classification scheme of the security approaches is schematized in Fig. 14. Moreover, we provide in Table 9 a summary of the main approaches that tackle security challenges in federated learning and highlight the criteria (proposed in Section 6.3) that each underlying approach satisfies.

6.1 Untargeted (Byzantine) Attacks

As mentioned earlier, Byzantine attacks are those attacks that seek to degrade the performance of the training model as a whole without having any specific client or data instance as a target. Byzantine attacks have been widely investigated in the context of federated learning where a variety of approaches have been proposed to analyze them and propose appropriate defense mechanisms. These approaches can be classified under three major techniques, i.e., security analysis, statistics and autoencoders. We explain hereafter each of these techniques and discuss the approaches that were proposed under each technique.

Security Analysis: The approaches that fall under this category seek to analyze the behavior of Byzantine attackers, discuss their impacts and examine the efficiency of some of the existing defense strategies against them. For example, in [154], the authors discuss the vulnerability of federated learning towards a class of poisoning attacks referred to as *model poisoning*. This attack can be performed using model replacement in which one or a group of colluding attackers attempt to substitute the global shared model with a malicious model, causing the model to misclassify future inputs. The authors discuss as well the potential defense strategies against such attacks and argue that they can be easily broken by attackers. Specifically, the authors argue that, using secure aggregation by the central server to filter out anomalous contributions, contradicts with the main idea of federated learning, which consists of capitalizing on the diversity of the clients in terms of non-iid training data including uncommon or low-quality data. Therefore, discarding the local model updates that diverge from those of the global model would not be logical. The difference between model poisoning and adversarial transformation attacks is also explained. While adversarial transformations take advantage of the boundaries between the model representations and those of the different classes to generate input data that tend to be misclassified, model poisoning seeks to maliciously manipulate and alter these boundaries to cause certain input data to be misclassified. Thus, the severity of model poisoning attackers stems from their ability to cause the model to misclassify certain inputs without having to modify these inputs. In [145], the authors explore Byzantine attacks on distributed machine learning. They first demonstrate that a single Byzantine attacker can compel the aggregation server to choose any arbitrary parameter vector, even one that is quite far in direction from

15. In the rest of the paper, the terms *untargeted attacks* and *Byzantine attacks* are used interchangeably.

TABLE 9: Summary of the approaches that address security concerns

Approach	Challenge	Technique	Main Idea	Criteria
Chen et al. [118]	Untargeted Attacks	Security Analysis	Analyze the vulnerability of federated learning towards model poisoning attacks that are performed using model replacement and demonstrate that some potential defense strategies against such attacks can be easily broken by attackers.	Criteria #3 and #5
Blanchard et al. [145]	Untargeted Attacks	Security Analysis	Demonstrate that a single Byzantine attacker can compel the aggregation server to choose any arbitrary parameter vector and show that a squared-distance-based aggregation rule is not efficient when more than one Byzantine attacker collude together.	Criterion #3
Fang et al. [146]	Untargeted Attacks	Security Analysis	Discuss a new type of model poisoning attacks in which the attacker takes advantage of the optimization problem to tune the local models on compromised clients in such a way to make the aggregate global model to maximally go in the direction that is opposite to the direction toward which the global model would have converged in case no attack took place.	Criterion #5
Munoz et al. [147]	Untargeted Attacks	Statistics	A Hidden Markov Model that assesses the similarity between the individual updates and the aggregate global model to identify and discard the malicious model updates at each training iteration.	Criterion #4
Blanchard et al. [145]	Untargeted Attacks	Statistics	A Byzantine-resilient federated learning approach that forces the vector output selected by the server to point, on average, to the same direction as the gradient.	Criterion #4
Li et al. [148]	Untargeted Attacks	Statistics	An SGD variant method that is resilient to Byzantine clients that arbitrarily alter the messages (generated by themselves) prior to sending them to the central server.	Criteria #4 and #7
Li et al. [149]	Untargeted Attacks	Autoencoders	A pre-trained autoencoder model that runs at the server's level to recognize anomalous model weight updates and identify their issuers.	Criterion #5
Bhagoji et al. [150]	Targeted Attacks	Security Analysis	Analyze a set of strategies that enable to carry out and counter targeted model poisoning attacks on federated learning and design an alternating minimization problem that takes into account both model poisoning and stealth metrics to enable attackers to escape detection.	Criterion #5
Fung et al. [151]	Targeted Attacks	Statistics	A defense system against Sybil-based poisoning attacks that adapts the learning rates of the clients based on the similarity of their gradient updates.	Criteria #3 and #4
Li et al. [152]	Targeted Attacks	Statistics	A spectral anomaly detection technique that capitalizes on the low-dimensional embeddings of the model updates to identify and eliminate the malicious ones.	Criterion #4

the other vectors. Thereafter, they show that a squared-distance-based aggregation rule, which selects the parameter vector that minimizes the sum of the squared distances to every other vector, is efficient in dealing with only one Byzantine attacker but becomes inefficient when two or more attackers collude together. In [146], the authors propose a new type of model poisoning attacks in which the attacker employs an optimization model to tune the local models that are supposed to be solved on the compromised clients. The objective is to make the aggregate global model to go in an opposite direction toward which it would have went in case no attack took place. The authors prove that the existing defense systems against model poisoning attacks in federated learning show limited performance in the presence of such a type of attacks.

Statistics: Statistics consists of a set of methodologies and techniques to collect, examine, analyze and come up with conclusions from data. In the context of federated learning security, statistical methods have been employed to examine the model updates generated by the clients, analyze their similarity/dissimilarity and draw appropriate conclusions on the existing/inexistence of Byzantine attacks. For example, the authors of [147] propose a Byzantine-robust federated learning approach called *Adaptive Federated Averaging*. The proposed solution leverages a Hidden Markov Model to identify and discard malicious model updates at each iteration. This is accomplished through assessing the similarity between the individual updates and the result yielded by the aggregate model. Malicious clients are then blocked from the federated learning system to improve future iterations. The authors of [145] design *Krum*, a Byzantine-resilient distributed machine learning approach. The main idea of *Krum* is to force the vector output selected by the server to (1) point, on average, to the same direction as the gradient and (2) have statistical moments bounded above by a homogeneous polynomial in the moments of a correct estimator of the gradient. The authors of [148] study a scenario in which in which Byzantine clients arbitrarily alter the messages (generated by themselves) prior to sending them to the central server. They then propose a method called Representational Similarity Analysis (RSA), which is a variant of the stochastic gradient descent method that is resilient to the considered Byzantine attack. The authors prove that the proposed method converges to a near-optimal solution at an $O\ 1\ k$ convergence rate.

Autoencoders: An autoencoder is a special class of neural networks which reproduces the input values into the output values. The primary interest of autoencoders are the hidden core layers rather than the output layers. It hence belongs to unsupervised learning and requires no target variables. In autoencoders, a number of neurons in the hidden layers that is lower than that of the input layers means that the hidden layers will elicit the essential information of the input values, learn most of the patterns of the data and disregard the *noises*. The main applications of autoencoders are dimensionality and noise reduction. Compared to the conventional dimensionality reduction approach of Principal Component Analysis (PCA) [155] which employs linear algebra to achieve its purpose, autoencoders excel at dealing with situations wherein data problems are non-linear and complex. In simple words, the mission of an autoencoder is to encode the data into a smaller version (compression) and then decode

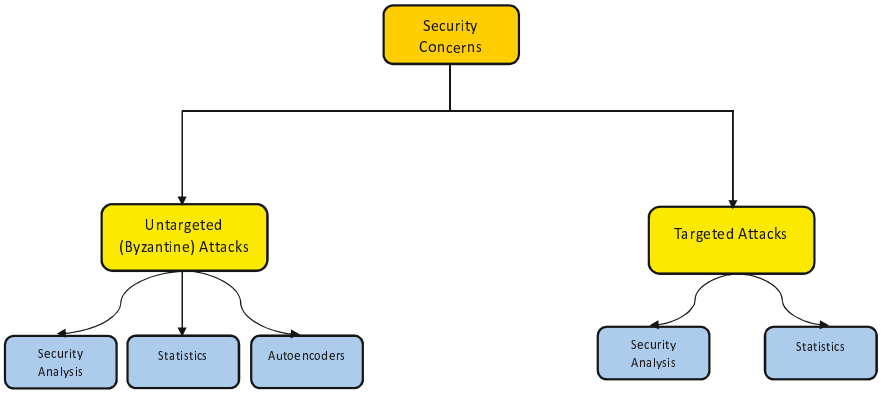


Fig. 14: Classification of the security-oriented approaches in federated learning

it back to regenerate the input (uncompression). While doing so, the autoencoder learns the features of normal data, compresses them into smaller-sized features and finally decodes them back into the input with a small error. Thus, when fed with anomalous data, an autoencoder fails to reproduce them and generates a large error, given that it is trained to reproduce normal data solely. Thus, in order to employ autoencoders for anomaly detection, it is sufficient to compute the error (mostly the Mean Square Error) of the output compared to that of the input and compare this error to a predefined threshold. Inspired by this idea, the authors of [149] propose a detection method for Byzantine attackers that capitalizes on a pre-trained anomaly detection model. In more detail, a pre-trained autoencoder model that runs at the server's level is employed to recognize anomalous model weight updates and identify their issuers. Accordingly, a credit score is assigned to each client based on its anomaly level. This score is used to weigh the contributions of each client when deriving the aggregate global model.

6.2 Targeted Attacks

Unlike Byzantine attacks which aim to deteriorate the overall performance of federated training, targeted attackers have specific malicious objectives and employ more sophisticated techniques to accomplish them. Technically speaking, they seek to modify the behavior of he training model on some particular data instances, while keeping the performance on the rest of instances uninfluenced. Targeted attacks on federated learning can be classified into two types: *data poisoning attacks* and *model poisoning attacks*. Data poisoning attackers aim to inject malicious data into the training dataset before the beginning of the learning process. The learning process is deemed to stay sound in the case of data poisoning attack. In federated learning, since an attacker has control over only its own device's data, data poisoning attacks have limited success and consequently received less attention from security community. On the other hand, model poisoning attackers aim to sway the learning model itself instead of compromising the data. Specifically, such attackers seek to make the collaboratively trained global model to misclassify a collection of chosen inputs with high confidence, while ensuring a high-quality convergence of the global model on the validation and test sets to avoid being detected. Technically speaking, such attackers produce their updates through optimizing for a malicious objective that aims at producing targeted misclassification, while also trying to negate the combined effect of the honest updates provided by the benign clients. Several techniques have been proposed in the literature to counter targeted attacks in federated learning, where these techniques can be classified into two major categories, i.e., security analysis and statistics. In the following, we explain each of these techniques in detail and discuss the approaches that have been proposed under each technique.

Security Analysis: The approaches that fall under this category seek to analyze the behavior of the attackers, discuss their consequences and examine the efficiency of some of the existing defense strategies in countering them. For example, the authors of [150] analyze a set of strategies that enable to carry out and counter targeted model poisoning attacks on federated learning. First, the explicit boosting strategy in which malicious participants try to negate the combined effect of benign clients is investigated. Thereafter, stealth notions, of which attackers can take advantage to improve their attack success chances such as weight update statistics, are investigated. An alternating minimization formulation problem that takes into account both model poisoning and stealth metrics is then proposed to enable attackers to escape detection. Finally, the authors empirically demonstrate that standard dirty-label data poisoning attacks, in which input data are manipulated to include adversarial examples, are not effective in federated learning, even when the number of adversarial examples is equal to that of local training data on each participating client.

Statistics: As is the case in the context of Byzantine attacks, in targeted attacks, statistical methods are used to analyze the model updates generated by the clients and study their similarity/dissimilarity to detect potential attacks. For example, in [151], the authors discuss a Sybil-based poisoning attack on federated learning and propose a defense system called *FoolsGold*. The considered attack can take two forms. The first is through label-flipping, where the labels of training examples pertaining to a particular class are flipped while keeping the data features unmodified. The second is through backdooring, where single features or small regions of the training data are augmented with an undercover pattern and accordingly relabeled. The main idea of *FoolsGold* stems from the observation that a set of Sybil attackers tampering a shared learning model will submit updates towards a single specific malicious objective, thus showing a similar behavior that is usually different from that of honest clients. Inspired by this perception, the authors propose to adapt the learning rates of the clients on the basis of their contribution similarity. More specifically, the learning rate of clients submitting unique gradient updates is maintained, while that of clients constantly submitting similar-looking updates is reduced. The authors of [152] are interested in detecting targeted attacks whose objective is to change the behavior of the model on some data instances. To do so, they put forward a spectral anomaly detection technique which capitalizes on the low-dimensional embeddings of the model updates to identify and eliminate the malicious ones. The intuition is that, in a low-dimensional latent feature space, the primary features of abnormal model updates would be radically different from those of the normal ones.

6.3 Desirable Criteria for Future Solutions

Based on the above classification and discussions, we identify a set of criteria that we believe are important to consider when designing future security solutions for federated learning. In what follows, we first present these criteria and then discuss how they can be practically capitalized on to craft efficient security solutions in federated learning.

Criterion #1: Make sure that the clients are honestly using their own data and not bogus data to train their local models.
Criterion #2: Account for Sybil-based attacks in which malicious clients can join the training process under multiple distinct identities to increase the impacts of their attacks.
Criterion #3: Account for collusion attacks in which two or more clients join forces to cause more significant damage to the training performance.
Criterion #4: Fight against intelligent attackers that adaptively limit their poisoned updates and intelligently add noise to their updates to escape detection.
Criterion #5: Fight against intelligent attackers that arbitrarily alter the weights of their local models and/or embed some defense avoidance models into their loss function during training.
Criterion #6: Decouple the security solution from the ability to access clients' training data or their submitted model updates.
Criterion #7: Be designed to operate in a non-IID data setting.

Several additional aspects need to be taken into consideration while designing prospective security solutions in federated learning. First, there is a need to come up with techniques to verify whether clients are honestly using their own data and not some fraudulent data to train their local models. Blockchain (explained later in the context of privacy concerns) offers a great potential to address this challenge. A second research direction would be to consider attacks that are performed in a colluding fashion. Most existing approaches consider attacks that are performed in an individual fashion. However, it is no secret that attackers can collaborate together to perform more painful attacks. Therefore, approaches that are tailored for collusion-based attacks need to be designed. Another interesting research direction would be to design security solutions that take into account intelligent attackers that try to trick the security countermeasures. Such attackers might, for example, adaptively limit their poisoned updates and/or incorporate defense avoidance models into their loss function optimization model to escape detection. Therefore, the design of the security solutions should account for such types of tricky behaviors. Another important prospective aspect would be to design the security solutions in such a way to align with the privacy requirements on the clients' data and model updates. More specifically, a large part of the existing security solutions require access to either clients' training data or their submitted model updates to detect the malicious behavior. This latter, although might be effective from the security perspective, contradicts with the privacy premises that the whole idea of federated learning relies on. Therefore, it would be of prime importance to come up with a trade-off between having an efficient security solution and preserving the privacy of the clients.

7 Privacy Concerns (Challenge 5)

In this section, we discuss the privacy concerns in federated learning, where we first classify the existing approaches based on the privacy challenge that they tackle, resulting in two categories: (1) differential privacy and (2) secure aggregation. Thereafter, under each of these categories, we provide a fine-tuned classification of the existing approaches based on the technique used to address the underlying challenge. It is worth noting that the adversary model considered in most of the existing privacy-oriented approaches assumes that all parties are honest-but-curious in the sense that they: (i) fully follow the protocol without manipulating it; (ii) do not engage in any collusion scenario with one another; but (iii) try to infer as much sensitive information as possible from the other participants. The classification scheme of the privacy-oriented approaches is schematized in Fig. 15. Moreover, we provide in Table 10 a summary of the main approaches that tackle privacy challenges in federated learning and highlight the criteria (proposed in Section 7.3) that each underlying approach satisfies.

7.1 Differential Privacy

Differential privacy is a mathematical concept that is used in the scope of statistical machine learning to fight against differential attacks and ensure that the processes of collecting, aggregating and analyzing data do not expose sensitive information on individual users [170], [171], [172]. Numerous techniques have been proposed in the literature to address differential privacy-related issues in federated learning. These techniques can be classified into three major categories, i.e., privacy analysis, noise injection and data-driven solutions. In the following, we explain each of these techniques in detail and discuss the approaches that were proposed under each technique.

Privacy Analysis: The approaches that fall under this category are proposed to analyze the privacy vulnerabilities of the federated learning paradigm and examine the efficiency of some existing defense strategies. For example, the authors of [156] investigate sensitive data leakage in federated learning with focus on logistic regression models. Two training approaches are considered in the analysis, i.e., synchronous and asynchronous. In the synchronous approach, the client computes gradients based on its own data in current batch. In the asynchronous approach, the client employs several batches to compute the gradients. The authors mathematically show that honest-but-curious clients can readily deduce the whole training data of other clients if the synchronized approach is adopted. On the other hand, the authors show that honest-but-curious clients can deduce nothing but some constraints of other clients' training data if the asynchronous approach is adopted. The authors of [157] examine several inference attacks on federated learning. The obtained results suggest that the leakage of unintended features through sharing the model updates exposes the federated learning paradigm to serious active and passive inference attacks. Such attacks enable malicious clients to infer both *memberships* (i.e., the presence of some data points in other clients' training data) and *properties* that depict some subsets of the training data (which are independent from the properties that the joint model seeks to determine). Additionally, further experiments reveal that common defense strategies such as dimensionality reduction, selective gradient sharing and dropout cannot prevent such inference attacks.

Noise Injection: The main idea of noise injection approaches is to allow the clients to add some noise to their gradient updates to prevent the server from using the actual gradients to infer sensitive information from the training data. Inspired by this idea, the authors of [158] propose an approach to counter differential attacks that seek to infer a client's contribution to the training process through analyzing the distributed training model. Thus, a differential privacy-preserving framework is proposed to prevent a learned model from exposing information leading to uncover whether or not a certain client participated in the distributed training of the model. The solution involves two principal steps, i.e., random sub-sampling and distortion. In the random sub-sampling step, a subset of the available clients is sampled and selected to receive the central model parameters and compute the model parameter updates on their local data. The distorting step consists of adding some noise to the sum of all scaled updates to prevent clients' crucial information from being leaked. The authors of [88] propose to enable clients to add some noise to their gradients prior to sending them to the server. Consequently, the

TABLE 10: Summary of the approaches that address privacy concerns

Approach	Challenge	Technique	Main Idea	Criteria
Li et al. [156]	Differential Privacy	Privacy Analysis	Demonstrate that an asynchronous communication approach in which the client computes the gradients over several batches is better in terms of clients' training data privacy in the presence of honest-but-curious participants.	Criterion #6
Melis et al. [157]	Differential Privacy	Privacy Analysis	Prove that sharing model updates enables malicious clients to infer both *memberships* (i.e., the presence of some data points in other clients' training data) and *properties* that depict some subsets of the training data (which are independent from the properties that the joint model seeks to determine).	Criteria #2, #3, #4 and #5
Geyer et al. [158]	Differential Privacy	Noise Injection	A distorting mechanism that adds some noise to the sum of updates to prevent inferring each client's contribution to the training process.	Criterion #3
Agarwal et al. [88]	Differential Privacy	Noise Injection	A Binomial mechanism that enables clients to add some noise to their gradients prior to sending them to the server and the server to estimate the noise in the model aggregation phase.	Criterion #7
Peterson et al. [159]	Differential Privacy	Noise Injection	A privacy-preserving federated learning framework that enables each client to train both a general model (with noise) and a private domain model and uses the mixture of experts technique to combine the outputs of both models.	Criterion #3
Liu et al. [160]	Differential Privacy	Noise Injection	A sketching method that is applied on the updates exchanged between the clients and the server to protect the clients' identities from being exposed.	Criterion #10
Triastcyn et al. [161]	Differential Privacy	Machine Learning	Train a GAN in a federated fashion on each client's data to generate artificial data that can replace the client's original data.	Criteria #5 and #12
Han et al. [162]	Differential Privacy	Machine Learning	A transfer learning framework that aims to improve the robustness of the federated training against the noise used as part of the differential privacy through enabling the clients to elect the most informative subset of their training data as trusted instances and train the model on these instances to compel the model to agree with the trusted instances.	Criterion #10
Hardy et al. [163]	Differential Privacy	Homomorphic Encryption	A three-party federated logistic regression model over messages encrypted with an additively homomorphic scheme.	Criterion #3
Mandal et al. [164]	Differential Privacy	Homomorphic Encryption	An additive homomorphic encryption mechanism that enables the server and clients to run a shared local gradient computation model and compute two additive shares of the local gradient on clients' data.	Criteria #1 and #13
Feng et al. [165]	Differential Privacy	Homomorphic Encryption	A bilateral privacy-preserving federated learning model that protects the privacy of not only client's raw training data but also model iterations and final model parameters using encryption.	Criteria #1 and #13
Bonawitz et al. [166]	Secure Aggregation	Secure Multiparty Computation	An SMC protocol that aims to protect the federated learning framework from honest-but-curious attackers through solely revealing the sum of model parameter updates to the server, only after a certain number of updates has been carried out.	Criterion #1
Bonawitz et al. [167]	Secure Aggregation	Secure Multiparty Computation	A double-masking strategy that enables each client to sample an additional random value to mask its model updates in order to prevent honest-but-curious servers from gaining access to these updates even in cases where the server can reconstruct the client's perturbations.	Criterion #1
Niu et al. [94]	Secure Aggregation	Secure Multiparty Computation	A secure federated submodel mechanism that relies on the concepts of secure aggregation, randomized response and Bloom filter to provide each client with a customized plausible deniability against the position of its submodel.	Criteria #8 and #10
So et al. [23]	Secure Aggregation	Secure Multiparty Computation	An additive secret sharing mechanism that adds some randomness into each local model, where this randomness is designed in such a way to vanish once the models are aggregated.	Criterion #13
Truex et al. [168]	Secure Aggregation	Secure Multiparty Computation	Integrate SMC into differential privacy to decrease the expansion of noise injection when the number of clients increases.	Criteria #10 and #15
Kim et al. [169]	Secure Aggregation	Blockchain	A blockchained federated learning architecture in which the aggregation is done in a decentralized fashion and the model updates get verified by specialized devices enjoying high energy levels.	Criterion #10

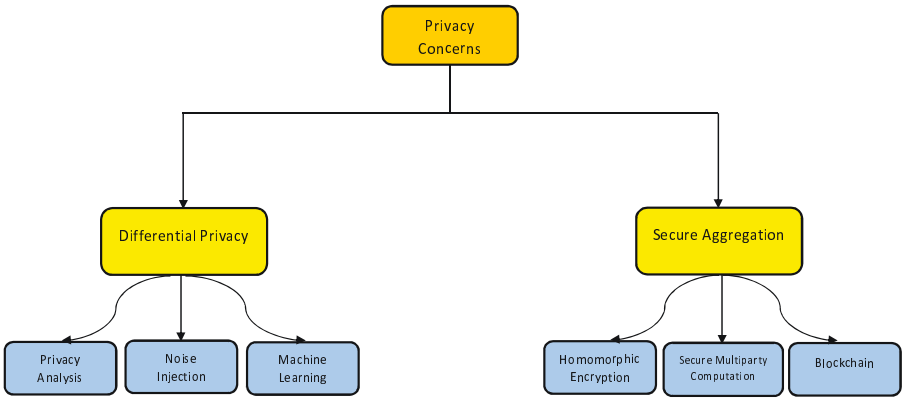


Fig. 15: Classification of the privacy-oriented approaches in federated learning

gradients' aggregation at the server would generate an estimate with noise that is equal to the sum of the noise added by each client. The main purpose of this approach is to tackle cases wherein clients do not trust the server to add the noise itself. From the technical perspective, the authors investigate a Binomial mechanism and show that such a mechanism attains almost the same utility as the Gaussian mechanism, while necessitating less representation bits. In [159], the authors propose a privacy-preserving federated learning framework, wherein each client trains both a general model and a private domain model. Differentially private (with noise) stochastic gradient descents are designed for the general shared model, while ordinary stochastic gradient descents are employed for each private domain model. Then, the Mixture of Experts (MoE) neural networks technique is employed by each client to combine the outputs of both models, in such a way to make each of the general and private models differently influence the predictions on each individual data instance. The authors of [160] aim to achieve a trade-off between the privacy and accuracy of the federated learning process using sketching algorithms. Specifically, the authors propose to apply sketching on the updates exchanged between clients and parameter server to protect the identities of private users from being exposed. The objective is to achieve local privacy when raw user information are not shared and only model updates are shared with a third-party after being protected, while at the same time maintaining the accuracy of traditional federated learning models.

Machine Learning: Machine learning solutions are proposed to provide an alternative to the complex mathematical privacy-preserving solutions that often entail high computation and communication cost to be implemented. A variety of machine learning solutions such as GANs, clustering and transfer learning have been employed for privacy-preserving purposes. For example, the authors of [161] propose *FedGP*, a federated learning approach that capitalizes on GANs to improve the privacy of the training process. The main idea is to train a GAN on each client's data to generate artificial data that can replace the client's original data. The GAN is trained in a federated fashion, where at the beginning of each iteration, the server transmits an (updated) common generator to all clients. Clients, at each iteration, update their models and submit generator updates to the server. Differential average-case privacy is then applied to protect the model against model inversion attacks. In such attacks, attackers examine the output probabilities of a target model for a particular class and then carry out the gradient descent method on an input reconstruction in an attempt to infer typical representations of a specific target. The authors of [162] aim to propose a federated learning model that is robust to systematically corrupted noisy data that are distributed across multiple agents as part of the differential privacy process. To do so, a collaborative privacy-preserving transfer learning framework is advocated. In this framework, clients serve as teachers for the central server (the student of the framework) whose role is to passively receive updates from the teachers. Teachers elect the most informative subset of their training data as trusted instances and train the model on these instances based on a consensus optimization method to compel the model to agree with the trusted instances. In this way, the clients learn to add changes of limited magnitudes into the data, which helps improve the performance of the federated model in the presence of noisy data.

7.2 Secure Aggregation

Secure aggregation refers to the problem of computing a multiparty sum of values without having any party reveal its individual value. In federated learning, secure aggregation refers to the problem of computing the sum of the local gradient updates of the clients without revealing the contribution of each client to any other party. Three main techniques have been employed in the literature to perform secure aggregation in federated learning, namely *Homomorphic Encryption*, *Secure Multiparty Computation* and *Blockchain*. In the following, we discuss each of these techniques and shed light on the approaches that capitalize on each particular technique.

Homomorphic Encryption: Homomorphic Encryption (HE) is a type of encryption that enjoys a supplementary evaluation ability to perform computation over encrypted data without having to access the *secret keys*. It enables computation functions to be executed directly on encrypted data while achieving the same (encrypted) results as if the functions were run on plaintext. To better clarify the idea, let's take the example of a medical research institution that wishes to conduct descriptive statistics on a population of Covid-19 patients at a certain hospital. The challenge in this scenario stems from the fact that the hospital cannot share its private medical records with the research institution due to the Health Insurance Portability and Accountability Act (HIPAA) privacy regulations. To overcome this challenge, the hospital can use HE to encrypt its medical records prior to sending them to the research institution. In this way, the medical records will be fully private and protected. The research institution then carries out its analytical operations on the encrypted data, downloads the encrypted output and decrypts it to uncover the plaintext answer. From the technical perspective, similar to the other encryption forms, HE employs a public key to encrypt data and permits only the parties that have the matching private key to have access to the unencrypted data. The main difference between HE and the other encryption methods is that HE capitalizes on an algebraic system to enable computations to be done on the encrypted data. Three types of homomorphic encryption can be distinguished, i.e., (1) partial homomorphic encryption which keeps sensitive data secure by only allowing selected mathematical functions to be applied

on the encrypted data; (2) somewhat homomorphic encryption which upholds limited operations that can be carried out only a pre-defined number of times; and (3) full homomorphic encryption which keeps information both secure and accessible. In the context of federated learning, HE is used to encrypt the messages exchanged between the server and the clients to prevent honest-but-curious participants from exploiting these message to infer sensitive information from clients' local data. For example, the authors of [163] address the challenge of private federated learning wherein (1) data are vertically divided across clients by features; (2) target variable is known by only one party; and (3) entities' data are not linked across clients. To this end, a three-party federated logistic regression model over messages encrypted with an additively homomorphic scheme is proposed. The authors show that the proposed model is secure against honest-but-curious clients and provide a formal analysis on the impact of entity resolution which results from joining together vertically split datasets using unique identifiers on the federated training accuracy. In [164], the authors aim at guaranteeing both data and model privacy in federated learning. Toward this end, they propose two privacy-preserving protocols for multi-party regression training. The first protocol is tailored for linear regression while the second is deigned for logistic regression. First, the server and clients run a shared local gradient computation model, allowing them to compute two additive shares of the local gradient on the clients' data following an additive homomorphic encryption. The objective is to prevent input leakage, even in extreme scenarios in which the user has only one data point. The server and active clients then apply an aggregation protocol to build one share of the global gradient. Thereafter, the server derives a second share of the global gradient using its local gradient shares. The authors of [165] propose a bilateral privacy-preserving federated learning model that protects the privacy of not only client's raw training data but also model iterations and final model parameters. Specifically, the authors propose to encrypt intermediate and final model parameters, thus allowing clients to train over a noisy global model, while ensuring that the sever obtains the exact updated model. The authors theoretically prove that their solution prevents honest-but-curious clients from getting local training data and local model updates from other clients even under collusion scenarios.

Secure Multiparty Computation: Secure Multiparty Computation (SMPC) is a subfield of cryptography that is interested in designing methods for entities to jointly compute a function over their inputs while keeping these inputs private. Different from the traditional cryptographic jobs which aim to ensure the integrity and security of the storage and communication in situations wherein the adversary does not belong to the network of participants, SMPC aims to protect the participants' privacy from each other. In federated learning, SMPC is used to allow the server to compute the sum of model updates coming from a large number of client devices without learning each client's individual contribution. For example, the authors of [166] propose an SMC protocol for the federated learning framework called *Secure Aggregation*. Secure Aggregation capitalizes on encryption to keep clients' local updates secret from the parameter server. The proposed protocol is intended to protect the federated learning framework from honest-but-curious attackers through solely revealing the sum of model parameter updates to the server, only after a certain number of updates has been carried out. The protocol consists of four rounds, where at each round the server collects messages from all clients and computes, out of these messages, an independent response to be sent to each client. In the first two rounds (*Prepare* phase), shared secrets are initiated. In the third round (*Commit* phase), each client submits encrypted masked model updates to the server, which piles them up. In the last round (*Finalization* phase), clients expose cryptographic secrets to enable the server to disclose the aggregated model updates. In [167], the authors propose a secure aggregation scheme to protect the privacy of each client's model gradient while embedding a collection of robustness and efficiency metrics into the design of the solution. The main component is a double-masking strategy that enables users to sample an additional random value to mask its model updates. The purpose is to prevent honest-but-curious servers from gaining access to the clients' updates even in harsh scenarios wherein the server is able to reconstruct the client's perturbations. The authors of [94] first propose a federated submodel learning framework in which clients need to download and upload only some relevant parts of the full model. They then address the privacy concerns that arise from such an approach, where the client needs to inform the server about the position of her submodel. To tackle this concern, a secure federated submodel mechanism is advanced. The secure mechanism relies on the concepts of secure aggregation, randomized response and Bloom filter to provide each client with a customized plausible deniability against the position of her submodel. In [23], the authors propose an additive secret sharing mechanism which adds some randomness into each local model to protect clients' privacy. This randomness is designed in such a way to vanish once the models are aggregated. The authors of [168] argue that the approaches that use differential privacy only end up with low accuracy in the presence of a large number of clients, each of which holding a small amount of data. To address these problems, they propose to integrate SMC into the differential privacy to decrease the expansion of noise injection when the number of clients increase while maintaining a certain level of reliability.

Blockchain: A blockchain is a time-stamped series of immutable data blocks that is managed by a cluster of computers which are not owned by any single entity. Each of these blocks of data (i.e., block) is secured and bound to each other using cryptographic principles (i.e., chain) [173]. The blockchain is a democratized system in the sense that it does not rely on any central authority. Technically speaking, the blockchain is a simple yet smart way of communicating information from one party to another in a fully automated and secure fashion [174]. One party of a transaction starts the process through creating a block, where this block get verified by a large number (i.e., millions) of computers distributed around the World. The verified block then gets added to a chain, creating a unique record with a unique history [175]. In such a system, falsifying a single record would mean falsifying the entire chain in millions of instances, which would be virtually impossible. In the context of federated learning, blockchain is used to decentralize the global aggregation process through enabling the blockchain network to exchange client' local model updates while verifying them. Blockchain is useful to both protect the individual local model updates from being exposed and verify the legitimacy of these updates. For example, the authors of [169] propose a blockchained federated learning architecture, which consists of miner and normal devices. Miners are selected devices which enjoy high levels of energy. Each normal device computes its local model update and forwards it to its corresponding miner in the blockchain network. Miners exchange and verify all the local model updates. They then run the Proof-of-Work (PoW), which allows them to create a block that stores the verified local model updates. This block is then added to a blockchain entity called distributed ledger to be downloaded by the different devices. This would enable each device to derive the global model update from the new block.

7.3 Desirable Criteria for Future Solutions

Based on the above classification and discussions, we identify a set of criteria that we believe are important to consider when designing future privacy solutions for federated learning. In what follows, we first present these criteria and then discuss how can they be practically capitalized on to craft efficient security solutions in federated learning.

> **Criterion** #1: Prevent honest-but-curious participants from deriving the gradients of the other participants through com-

puting the difference between two consecutive global joint models.

Criterion #2: Prevent attacks that aim to infer properties that only hold for a subset of the training inputs but not for the class as a whole, i.e., properties that are independent of the class's features (e.g., inferring that Alice appears in some of John's photos when we are using John's photos to train a gender classifier).

Criterion #3: Prevent inference attacks that aim to know whether or not a certain client has participated in a certain training iteration.

Criterion #4: Consider passive attacks wherein attackers examine the updates and carry out inference without performing any changes to the local and global training processes.

Criterion #5: Consider active attacks that carry out additional local computations and submit the resulting values to the global model to perform their inference attacks.

Criterion #6: Investigate the impact of the model's hyper-parameters, batch size, number of participant and communication mode (i.e., synchronous or asynchronous) on the likelihood of leaking sensitive information.

Criterion #7: Account for situations wherein the clients do not trust the server to apply the differential privacy measures (e.g., adding noises to the local gradients).

Criterion #8: Derive a trade-off between privacy-preserving techniques (i.e., noise injection and cryptography) and commununication efficiency.

Criterion #9: Derive optimal bounds in terms of noise ratio to be added to the individual gradients based on metrics such as communication overhead, data representativeness and inter-client data variance to avoid adding too much or too little noise.

Criterion #10: Find a trade-off between noise injection and training model's accuracy.

Criterion #11: Design adaptive privacy-preserving solutions that take into account the software and hardware heterogeneity of the client devices (e.g., some devices might afford computation with noise while others might not).

Criterion #12: Capitalize on the strength of GANs to generate and operate on artificial data that replace the client's actual data.

Criterion #13: Consider not only clients' data privacy but also training model's privacy to prevent competing businesses from learning other businesses' models and using them to gain competitive advantages.

Criterion #14: Tailor the privacy-preserving solution to be independent from the machine learning model used for training.

Criterion #15: Design hybrid solutions that combine the strengths of both differential privacy and secure aggregation to achieve a trade-off between inference privacy and training model accuracy.

The current literature on privacy in federated learning can be improved in many aspects. First, the privacy preservation solution should account for both active and passive inference attacks. Most of the existing approaches have focused on countering the passive inference attackers that carry out their attacks without entailing any change to the local and global training processes. On the other hand, active attackers might be harder to detect and fight against since they carry out additional computations and submit fabricated values to the global model to push it to reveal sensitive information. Therefore, more elaborate solutions need to be designed to deal with such attackers. A second research direction that needs further investigation is the protection of the training model's privacy. Most of the existing approaches focus on protecting clients' local data from being leaked to other clients or to the server, while ignoring the privacy of the training model itself. Yet, breaching the privacy of the training model might lead to painful consequences since it could enable clients working for some businesses to learn other businesses' (better) training models in order to use them on their own data to gain competitive superiority. A third research perspective would be to conduct methodological investigations on the impacts of several factors such learning hyperparameters, batch size, number of participants and communication mode (i.e., synchronous or asynchronous) on the probability of leaking sensitive information. Such investigations could lead to more advanced privacy solutions that can optimize many metrics in a simultaneous fashion.

Another research direction that is specific to the differential privacy would be to derive optimal bounds in terms of the noise ratio that needs to be added to the individual gradients. Deriving such bounds might be based on metrics such as communication overhead, data representativeness and inter-client data variance. This is important to derive adequate amounts of noise, thus avoiding to adding too much (which results in high communication overhead) or too little (which might make the privacy solution ineffective). One additional perspective to be considered in the future would be to adapt the privacy preserving solutions to the software and hardware heterogeneity of the client devices. This heterogeneity could enrich the privacy solution with valuable information and offer it a higher flexibility. For example, the devices that enjoy high hardware and software specifications could be assigned the heavy privacy solutions (e.g., noise injection) while the devices that enjoy lower hardware and software specifications could be assigned lightweight privacy solutions. An additional aspect that is worth investigating would be to capitalize on the strengths of data-driven solutions such as GANs, transfer learning and multi-task learning to design more lightweight privacy preserving approaches. In fact, data-driven solutions have shown great efficiency and have the potential to reduce the computation and communication overhead compared to the traditional privacy preservation solutions such as noise injection, SMPC and homomorphic encryption. Finally, it would be of prime importance to design granular privacy solutions at the levels of specific users or even data samples. In practice, the privacy constraints differ across devices and across data samples on a client's device. Therefore, designing granular device and sample-level rather than local or global model-level privacy preservation solutions would be an interesting future direction.

8 Service Pricing (Challenge 6)

Service pricing approaches are introduced to analyze the interactions between the server and client devices from an economical point of view. In fact, clients are usually mobile devices that are owned by rational users that tend not to accept to spend the resources of their devices running local training for a model that is owned by the server. Therefore, economic models are needed to incentivize clients to participate in the federated learning process. Toward this end, the existing approaches that are proposed under this category capitalize on game theory and mechanism design to model the economical interactions among clients (referred to as *service providers*) and server (referred to as the *model owner*). In the following, we discuss each of these techniques, i.e., *game theory* and *mechanism design* in detail and explain the approaches that were proposed in the literature under each technique. The classification scheme of the service pricing approaches is schematized in Fig. 7. Moreover, we provide in Table 7 a summary of the main approaches that tackle service pricing challenges in federated learning and highlight the criteria (proposed in Section 8.1) that each underlying approach satisfies.

Game Theory: Game theory is a formal study of conflict and cooperation in a multilateral interactive environment [182]. It offers mathematical tools for designing automated decision-making models

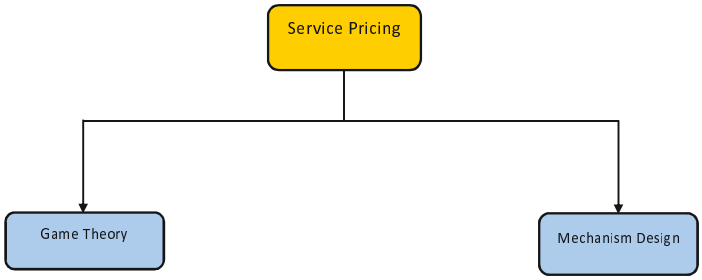


Fig. 16: Classification of the service pricing approaches in federated learning

TABLE 11: Summary of the service pricing approaches

Approach	Challenge	Technique	Main Idea	Criteria
Feng et al. [99]	Service Pricing	Game Theory	A Stackelberg game model in which the clients determines the price of their training data in such a way to convince the server to acquire more data from them.	Criteria #1, #2 and #5
Sarikaya et al. [176]	Service Pricing	Game Theory	A Stackelberg game model in which the model owner, at each gradient update iteration, offers clients an incentive in such a way to motivate them to assign larger computation power to their local training tasks.	Criteria #1 and #3
Song et al. [177]	Service Pricing	Game Theory	A Shapley value-based approach that assesses the contribution of each client through reconstructing the model on different combinations of the datasets using the intermediate results of the training.	Criteria #1 and #4
Zhan et al. [178]	Service Pricing	Game Theory	An Stackelberg game in which the server seeks to minimize the total reward given to the edge nodes and edge nodes aim to maximize their individual revenues.	Criteria #1 and #5
Pandey et al. [179]	Service Pricing	Game Theory	An incentive mechanism that relies on a two-stage Stackelberg game model which aims to jointly maximize the utility of the edge server and clients that interact via a third-party application platform.	Criteria #1 and #2
Jiao et al. [180]	Service Pricing	Mechanism Design	An auction mechanism that takes into consideration a variety of metrics such as clients' data sizes, non-IID degree of their data and their limited wireless spectrum resources to design and solve a social welfare maximization problem.	Criteria #1 and #2
Kang et al. [181]	Service Pricing	Mechanism Design	A contract theory approach that maps the amount of resources contributed by each client into some equivalent reward amount to incentivize clients to participate in the federated learning paradigm.	Criteria #1, #3 and #4

for rational agents in strategic scenarios. Such agents may represent individuals, machines, firms, software, or any combination of them. In other words, game theory aims at mathematically discovering the actions that a certain party (called *player* in game theory) should take in order to maximize its success chances. In federated learning, game theory is used to help the model owner and service providers derive pricing schemes for the training service on the basis of several criteria in such a way to maximize the profits of parties. For example, the authors of [99] address the challenges of motivating the participation of clients in the learning process. To do so, a service pricing scheme whereby clients decide on the price of one unit of their training data. Based on the decided prices, the server determines the size of training data that will be acquired from each client. Using this information, clients then adjust their prices in an attempt to convince the server to acquire larger training data sizes. This scenario is model as a Stackelberg game theoretical model. In [176], the authors propose an incentive mechanism that aims to motivate clients to assign larger computation power to their local training tasks. The ultimate objective is to accelerate the convergence through reducing the chances of encountering stragglers that take long time to submit their updates. The solution is modeled as a Stackelberg game in which the model owner (the leader of the game), at each gradient update iteration, offers clients a certain amount as an incentive. Based on the offered incentive, clients (followers of the game) determine the amount of Central Processing Unit (CPU) power that will be allocated to compute the gradient update. The equilibrium of the game is derived through deducing the average time needed to complete a single stochastic gradient descent iteration. The authors of [177] propose to employ Shapley value from coalitional game theory to assess the contribution of each client in the federated training. The goal is to come up with profit sharing strategies that take into consideration the contribution of each client to the joint model. The main intuition of the proposed solution is to reconstruct the model on different combinations of the datasets using the intermediate results of the training in such a way not to entail any extra training on the client devices. In [178], the authors design an incentive mechanism to motivate edge nodes to participate in the federated training. The problem has been formulated as a Stackelberg game in which the server seeks to minimize the total reward given to the edge nodes and each edge node aims to maximize its individual revenue. A deep reinforcement learning algorithm is also proposed to model the ambiguity of contribution evaluation through learning system states from historical training records. The authors of [179] propose an incentive mechanism that employs a two-stage Stackelberg game model which aims to jointly maximize the utility of the edge server and clients that interact via a third-party application platform.

Mechanism Design: Mechanism design is a branch of economic theory that aims to explore the mechanisms wherein a certain outcome can be achieved [183]. It helps understand how businesses may attain optimal outcomes when individual self-interest and incomplete infor-

mation comes into the picture. Similar to game theory, mechanism design is employed in federated learning to help the model owner and service providers derive pricing strategies for the training service. For example, the authors of [180] design an auction mechanism to model the interactions between clients and the federated learning platform. The pricing scheme takes into consideration clients' data sizes, the non-IID degree of their data (quantified using the earth mover's distance measure) and their limited wireless spectrum resources. Based on these metrics, a data quality function is designed using real-world datasets. Thereafter, a social welfare maximization problem is formulated based on the data quality function. To solve this problem efficiently, the authors put forward a deep reinforcement learning-based auction mechanism. The authors of [181] study the problem of incentivizing mobile devices to participate in the federated learning paradigm by the model owner. To do so, an approach that capitalizes on contract theory, a subfield of mechanism design wherein agreements are enforceable, is proposed to map the amount of resources contributed by each mobile device into some corresponding rewards. The profit function of the task publisher is defined in terms of total time of one global iteration of collaborative training minus the total amount of monetary rewards that should be given to mobile devices in recognition to their contributions. On the other hand, the profit function of each mobile device is defined in terms of rewards received from the task publisher minus the computation and communication costs entailed by the training process.

8.1 Desirable Criteria for Future Solutions

Based on the above classification and discussions, we identify a set of criteria that we believe are important to consider when designing future service pricing solutions for federated learning. In what follows, we first present these criteria and then discuss how can they be practically capitalized on to craft efficient service pricing solutions in federated learning.

Criterion #1: Provide incentives for clients to motivate them to participate in the training process.
Criterion #2: Investigate the impact of the training data size and quality on both the client devices' profits and training model's accuracy .
Criterion #3: Motivate clients to assign more computation power to the local training to improve the accuracy and avoid the problem of stragglers.
Criterion #4: Link the rewards given to the client devices with the degree of their contributions to the training and the quality of their model updates.
Criterion #5: Derive the payments to be given to the clients in such a way to satisfy their individual rationality (i.e., no loss to data owners from trading) without forcing model owners to make too much payment.
Criterion #6: Motivate data owners to truthfully reveal the size of their data and their computation capabilities.
Criterion #7: Attract more data owners with high-quality local training data to ensure efficient federated learning.
Criterion #8: Investigate open market approaches to reduce the aggressiveness of the competition among service providers and model owners.

The literature on service pricing in federated learning is still in its infancy, where many interesting aspects need to be explored and investigated. To begin with, we need comprehensive pricing schemes that take into consideration a variety of metrics such as the impact of the training data size and quality on both the client devices' profits and training model's accuracy. The pricing model should also account for the degree and quality of contribution of each single client on the training quality. In addition, the pricing scheme should be designed in such a way to incentivize clients to assign larger portions of computation power to the local training. This is important to ensure the continuity and stability of the federated training and hence reduce the chances of encountering stragglers. Furthermore, the current literature approaches employ non-cooperative game theory (e.g., Stackelberg games) to formulate and solve the service pricing problem. The shortcoming of this approach in federated learning arises from the nature of the data which can be shared and resold by clients to more than one model owner. This situation results in an extremely aggressive competition among service providers (i.e., clients) to sell their services even at lower prices and also among model owners themselves to acquire high-quality data from clients even at higher prices. To deal with such a situation, open market approaches that rely on a third-party platform to regulate the interactions among model owners and service providers are worth being investigated. For this purpose, approaches such as two-sided market theory and cooperative game theory would be interesting to explore.

9 Future Research Directions

We discuss in this section some future research directions that we believe are interesting to work on and investigate in the future. The research directions are classified based on the high-level challenges that they are intended to tackle.

9.1 Statistical perspective

We discuss in the following the future research directions that are related to the statistical challenges of federated learning.

9.1.1 Utility-based Federated Training

Most existing federated learning aggregation models treat clients' contributions equally. This results in unnecessary computation and communication overhead and might also degrade the quality and accuracy of the resulting global learning model in case of unreliable local data or poor local training models. Therefore, we believe that it would be interesting to design utility-based federated training models so as to assign a training utility value to each participating client. This utility would be used to quantify the appropriateness of the contributions provided by each client and could be based on a variety of criteria such as data size on the client's device, local computation time history and impact of the client's data on the non-IID degree of the overall data distribution. On important factor to consider here is that the utility model should be designed to be dynamic in the sense that it should change over the time to capture the changes that happen at the level of each client in terms of data quality, local device characteristics and client behavior.

9.1.2 Proactive Pre-training Data Heterogeneity Detection

The challenge of detecting data heterogeneity across client devices has been extensively studied in the context of federated learning. Nonetheless, most of the proposed solutions rely on metrics such as local dissimilarity, which can only be computed after the federated training has terminated. This makes these solutions less effective in improving the accuracy of the global learning model. Therefore, we urge the need to design proactive data heterogeneity detection solutions that could warn the server a priori about the degree of heterogeneity across the selected participant devices. Such solutions could be capitalized on to enable the server to revise its choices of clients, which is essential for reducing the number training iterations toward reaching the desired accuracy. To design such proactive solutions, we believe that system-level metrics of client devices such as storage, computational, and communication capabilities along with the training history of the device, which could be known before the start of the federated learning process, are worth capitalizing on to have an idea about the degree of heterogeneity across devices.

9.1.3 Model heterogeneity

Despite the unprecedented data privacy guarantees that federated learning promises to its participants, model privacy preoccupations might be the main stumbling-block preventing some businesses from adopting this emerging learning approach. More specifically, during the federated training, some participants might try to learn other competing participants' models and parameters in order to apply them to their internal prediction/classification tasks, and potentially gain some competitive advantages. One possible solution to such a case would be to foster model heterogeneity, where each client independently designs its own training model. Yet, the main challenge toward adopting model heterogeneity would be the difficulty of sharing information in a blackbox fashion among clients adopting different training models. Knowledge distillation might be an interesting solution to this problem as highlighted in [75].

9.2 Communication perspective

We discuss in the following the future research directions that are related to the communication efficiency challenges of federated learning.

9.2.1 Device-Oriented Event-Triggered Communication

Apart from the synchronous and asynchronous communication schemes that are widely adopted in federated learning, the workload-free nature of client devices makes it more realistic to think of event-triggered communication schemes. Unlike traditional worker nodes that are workload-dedicated and workload-ready, client devices in federated learning are general-purpose devices over which the server has no control. They also are not supposed to be running heavy computations. Indeed, traditional worker nodes in data centers are designed to always be ready to take their next job from the central node, directly after delivering the results of their current job. On the other hand, client devices do not have this property and might not be necessarily active at any given iteration of federated learning. That being said, it would be of prime importance to investigate device-oriented and event-triggered communication schemes. Such schemes should be designed in such a way to enable each device to take the decision on when to get activated and start interacting with the server in an event-triggered fashion.

9.2.2 Communication Reduction at the Pareto frontier

Several approaches have been proposed to improve the communication efficiency in federated learning. These approaches are mainly based on decreasing either the size of the exchanged model updates or frequency of communicating these updates. However, the benefits of applying such solutions often come at the price of loosing some of the prediction/classification's accuracy. Therefore, profound analysis on the trade-off between accuracy preservation and communication efficiency should be carried out to come out with effective and efficient communication models for federated learning. More specifically, we need new communication approaches that can prove to be efficient at the Pareto frontier, i.e., that can give an accuracy level that is larger than any other approach under the same communication budget and across a large domain of communication/accuracy profiles.

9.2.3 Partial Peer-to-Peer Updates Sharing

Sharing the whole set of (heavy) model updates with the server at each and every iteration, for a large number of iterations, is the main reason for communication bottlenecks in federated learning. Periodic and dynamic aggregation approaches [25], [58], [22] have been proposed in an attempt to decrease the communication overhead and avoid these bottlenecks. According to the periodic aggregation approach, clients are allowed to carry out several local iterations prior to uploading their updates to the server. In dynamic aggregation, the aggregation at the level of the server is performed only when judged necessary and some criteria are met. Despite the effectiveness of these ideas in terms of minimizing the communication rounds, applying them in practical federated learning scenarios might still result in bottlenecks at the level of the server per communication round, where the server is still supposed to receive a large number of heavy model weight updates during some iterations. Therefore, we believe that it would be interesting to investigate partial peer-to-peer model updates sharing schemes in which the clients synchronize the model updates with only a part of other workers instead of repeatedly sharing them with the server. Such a peer-to-peer communication approach can improve the communication efficiency and help avoid bottlenecks at the level of the server. This can be achieved by using the concept of clustering from machine learning, which can help uncover those groups of client devices that sharing the model updates among their members would have the best impact on the model's accuracy. The clustering might be done based on a variety of criteria such as devices' location, resource characteristics and trust relationships among the devices.

9.2.4 Multi-Hop Routing

Compared to single wireless links, multi-hop wireless networks can provide a wide range of benefits to the federated learning experience. First, it can increase the coverage of the network, thus bringing the federated training to a wider set of users in a wider set of areas and regions. Second, multi-hop networks allow for transmission over several short links instead of a long single link, which demands less transport energy and power. Third, with multi-hop networks, multiple paths are available where alternative paths can be used to improve the robustness and resilience of the network in case of any failure. Finally, multi-hop networks allow for higher data rates in the network which results in higher throughout. Despite all these advantages that multi-hop networks can bring to the federated learning, it has been quite under-investigated in the current literature. Specifically, we need to design approaches that can boost multi-hop federated learning over wireless communications, enabling several groups of clients to collaboratively participate in the federated training. Moreover, different from single-hop federated learning frameworks, the model updates in multi-hop federated learning have to pass through several noisy and interference-prone wireless links, thus leading to slower updates. Thus, it would be interesting as well to address this challenge and design innovative wireless multi-hop federated learning systems with high accuracy, warranted stability and fast convergence speed.

9.3 Client Selection and Scheduling

We discuss in the following the future research directions that are related to the client Selection and Scheduling challenges of federated learning.

9.3.1 Optimization of Interdependent Server-Client Strategies

Despite the abundance of optimization models that try to optimize the resource management at the client devices participating in the federated learning process, these models treat the strategies of the server and clients as being independent. Nonetheless, the strategies adopted by each of these parties strongly influence the welfare of the other party. For example, the amount of resources that the client decides to dedicate to the local training strongly influences the accuracy of the generated global model. Similarly, the decision of the server in terms of reward to be given to the clients versus their training service strongly influences the welfare of the clients. Therefore, considering the strategies of the server and clients independently would generate less thoughtful decisions. Thus, we argue that the strategies of the server and clients should be modeled in an interdependent fashion so as to enable each of these parties to make more informed decisions that consider the strategies of the other party as well as the best

responses to them. In this context, we believe that game theory is a perfect candidate to model the interdependencies between the server and clients' strategies. It offers a variety of tools and techniques that help model and analyze the interdependecy in the strategies such as best response, Nash equilibrium, maxmin, minmax, Stackelberg, hedonic and matching games.

9.3.2 Determining the Optimal Number of Clients Per Iteration

Determining the appropriate number of clients that need to participate in every federated learning iteration is crucial for the success of this emerging learning approach [123]. In fact, selecting a large number of clients might result in unnecessary communication overhead without bringing any improvement to the model's accuracy. Similarly, selecting a little number of clients might make the model to need more training iterations and thus to take longer time to converge. Therefore, we highlight the importance of designing models to determine the optimal number of clients that need to participate in each training iteration. Such models might take several constraints of into consideration such as the impact of the number of clients on the non-IID degree of the overall data, the reliability of the clients being invited to join and the impact of adding more clients on the waiting delay for model updates.

9.3.3 Client-to-Client Trust Establishment

The whole idea of federated learning presumes that the different involved parties, i.e., server and clients are trustworthy and willing to honestly do their assigned tasks. Nonetheless, in such an open and distributed environment in which the server has no control over the clients, and the clients act independently from one another, trust can no longer be assumed; it must be investigated. Although some trust (and reputation) models have been proposed in the context of federated learning, these models focus only on the trust relationships between the server and clients. Yet, we argue that it would be of prime importance to investigate the trust relationships among the clients themselves. Establishing client-to-client trust relationships would be a key enabler for many distributed communication architectures that have been or to be proposed to improve the communication efficiency such as periodic aggregation, dynamic aggregation, over-the-air-computation and partial peer-to-peer updates sharing.

9.4 Security Perspective

We discuss in the following the future research directions that are related to the security challenges of federated learning.

9.4.1 Fighting Against Collusion-based Attacks

Most of the security solutions that are proposed in the context of federated learning only consider attacks that are performed in an individual fashion. Yet, attackers are becoming smart enough to think of more sophisticated attack scenarios in which one single attack can be carried out by involving many client devices. The objective is to complicate the detection and prevention processes. Unfortunately, the existing security solutions that are tailored for individual attacks cannot be directly mapped nor easily adapted to collusion-based attacks. For example, a profound security analysis presented in [145] showed that squared-distance-based aggregation rules, which are used in many security solutions in federated learning, are not efficient when more than one attacker collude together.

9.4.2 Privacy-Preserving Security Assurance

A vast majority of the existing security solutions demand access to either clients' training data or to their submitted model updates to analyze and detect and fight against any suspicious behavior. This, however, contradicts with the privacy premises that the whole idea of federated learning is based on and threatens to break the confidentiality of clients' private data. Therefore, it would be of prime importance to decouple the future security solutions from the ability to access clients' training data or their submitted model updates. There should be a trade-off between having an efficient security solution and preserving the privacy of the clients' data.

9.4.3 Zero Trust Security

In a federated learning scenario, the edge servers come into contact with a huge number of client devices, which entails an unlimited number of security risks. There is plenty of devices of different sizes and types, from wearable devices to refrigeration and heating systems. This heterogeneity, along with the communication channels between the edge servers and client devices, gives attackers a non-negligible number of attack opportunities, most of which is not covered by the conventional security defenses. Traditionally, security administrators concentrate their security measures at the level of edge nodes, equipping the access points (e.g., routers and switches) with firewalls to filter the access to these nodes on the basis of some predefined protocols. Such a perimeter-based security approach cannot counter *lateral threat movement*, a set of techniques that attackers make use of, after acquiring initial access, to proceed deeper into the network to sneak into sensitive data and other invaluable assets. Traditional security solutions consider all the parties within the network to be trusted. Yet, if a federated learning system involves devices with minimal security measures, they could be used as weak points to acquire access to the central cloud servers. Therefore, a zero trust architecture would be preferable in such a case. More specifically, a zero trust architecture departs from the assumption that every node inside a network could be at some point hostile or compromised by external attackers and admits that the initial point of contact midst a cyberattack is often not the intended target. Zero trust puts in place a micro-segmentation process of the network (based on users, data types and applications) through dividing this network into smaller logic components in such a way that only authorized end-points could have access to the data and applications that are hosted in these components. This is advantageous over perimeter-based security in that smaller segments exhibit decreased attack surface and are easier to protect and manage. Thus, instead of investing all the defense mechanism at the main entry points, the zero trust architecture converts the whole federated learning network into a defense system that raises an alarm whenever an unauthorized party tries to perform an action. Consequently, even if the security measures of the outer edge/end device get breached, the chances that attackers will move far laterally across the network to acquire access to valuable assets would be minimal.

9.5 Privacy Perspective

We discuss in the following the future research directions that are related to the privacy challenges of federated learning.

9.5.1 Defining Optimal Bounds of Noise Ratio

Differential privacy is an indispensable component of most modern machine learning solutions to guarantee that the processes of collecting, aggregating and analyzing data do not expose sensitive information on individual users. One main component of differential privacy is noise injection, where clients add some noise to their model updates to prevent the server from revealing some sensitive information through analyzing their gradient updates. Nonetheless, in the current noise injection solutions, it is not clear how much noise should be added to the model updates. In fact, adding too much might result in high communication overhead, whereas adding too little noise might make the privacy solution ineffective in terms of privacy preservation. Therefore, we believe that it would be of prime importance to design models that can derive optimal bounds

in terms of noise ratio to be added to the individual gradients. Such solutions might take into consideration several criteria such as data representativeness, communication overhead and inter-client data variance.

9.5.2 Granular Privacy Solutions

The current privacy preservation solutions in federated learning are designed to be generic over all the client devices and data samples. However, in practice, the privacy requirements and constraints may largely vary between devices and even between the data samples on one single device. Therefore, it would be interesting to replace the current local or global model-level privacy solutions with granular device-level and even data sample-level ones. Nevertheless, we recognize and note the complexity of developing such granular solutions, owing to the huge number of client devices and gigantic volumes of data samples on these devices. Therefore, we believe that designing granular privacy solutions can be done in conjunction with some clustering solutions that group the devices and data samples according to some common characteristics. Consequently, granular solutions at the level of each cluster would be a good and more realistic alternative to the current generic privacy solutions.

9.5.3 Adaptive Privacy Solutions

One of the main requirements for successfully designing and applying granular privacy solutions is first to make the privacy model adaptive. Such adaptive models should provide a comprehensive analysis on the impacts of several factors related to federated learning such as data batch size, model's hyperparameters, number of participating clients, and communication mode on the likelihood of leaking sensitive information. It is also important for the prospective adaptive privacy solutions to take into account the software and hardware heterogeneity across the client devices. For instance, some devices are able to support heavy computation with noise, while others might not be able to afford heavy noise injection. Having such an analysis would help better understand the privacy requirements and constraints for each particular device (or a cluster of devices) and/or data sample (or a cluster of data samples) and would hence facilitate the design of granular privacy solutions.

9.6 Service Pricing

We discuss in the following the future research directions that are related to the service pricing challenges of federated learning.

9.6.1 Open Market-based Federated Learning

Service pricing is essential to guarantee the success and continuity of the federated learning approach. In fact, it is undeniably important to incentivize the clients to participate in the federated learning process through offering them some monetary rewards. The current literature on service pricing in federated learning capitalize on non-cooperative game theory to formulate the pricing problem. The disadvantage of this approach in federated learning stems from the nature of the data which can be shared and resold by clients to more than one federated learning model owner. This creates an extremely aggressive competition among clients to sell their training services even at lower prices and also among model owners themselves to acquire high-quality training from clients even at higher prices. To avoid these situations, we propose to study open market approaches such as two-sided market theory and cooperative game theory that capitalize on a third-party platform to regulate the interactions among model owners and clients to organize the interactions among them.

9.6.2 Multi-Objective Pricing

Federated learning is a completely open environment wherein participants are non-technical users who often do not have any interest in the success of the federated training process. Therefore, we argue that the pricing scheme should designed not only to motivate clients to participate in the training, but also to guarantee that providing high-quality data and training is the best option for the clients. In fact, the pricing scheme should be designed in such a way to incentivize the clients to assign larger portions of computation power to the local training. This is necessary to guarantee the continuity and stability of the federated training, thus decreasing the chances of encountering *stragglers*. The pricing scheme should also take into account the quality and size of the data at the clients' side and thus should be designed in such a way to attract more data owners with high-quality local training data.

10 Conclusion

In this survey, we propose a three-level classification scheme that categorizes the federated learning approaches, respectively, based on the high-level challenge they address, the sub-challenges within each high-level challenge category, and technique used to address each corresponding sub-challenge. In particular, we survey the statistical, communication efficiency, client selection and scheduling, security, privacy and service pricing challenges and break down each of these challenges into a set of fine-grained sub-challenges. Then, we explain and demystify the main techniques that are used to address each underlying sub-challenge. These techniques include, among others, data augmentation, active learning, multi-task learning, transfer learning, parameter tuning, knowledge distillation, weighted optimization, compression, periodic aggregation, over-the-air-communication, game theory, mechanism design, trust and reputation, multi-objective optimization, heuristics, reinforcement learning, statistics, autoencoders, noise injection, homomorphic encryption, secure multiparty computation and blockchain. For each category of high-level challenges, we provide a set desirable criteria that are aimed to help the research community design innovative and efficient future solutions. Finally, we survey and discuss the application of federated learning in the communication and networking field and shed light on potential future applications.

In all, we believe that our survey is the most comprehensive in terms of challenges and techniques it covers and the most fine-grained in terms of the multi-level classification scheme it presents. Moreover, it is the first survey that provides a set desirable criteria for each specific challenge of federated learning.

Acknowledgments

This work is partially funded by the Natural Sciences and Engineering Research Council of Canada (NSERC) under grant number *RGPIN-2020-04707*, by the Université du Québec en Outaouais (UQO), by the Lebanese American University, by Khalifa University of Science, Technology & Research (KUSTAR), by the European Union's Horizon 2020 research and innovation program under grant agreement n^o101016509 (project CHARITY), by the Academy of Finland 6Genesis project under Grant No. 318927, and by the Academy of Finland CSN project under Grant No. 311654.

References

[1] X. Chen, L. Jiao, W. Li, and X. Fu, "Efficient multi-user computation offloading for mobile-edge cloud computing," *IEEE/ACM Transactions on Networking*, vol. 24, no. 5, pp. 2795–2808, 2015.

[2] O. A. Wahab, J. Bentahar, H. Otrok, and A. Mourad, "Resource-aware detection and defense system against multi-type attacks in the cloud: Repeated bayesian stackelberg game," *IEEE Transactions on Dependable and Secure Computing*, 2019.

[3] O. A. Wahab, J. Bentahar, H. Otrok, and A. Mourad, "Optimal load distribution for the detection of vm-based ddos attacks in the cloud," *IEEE transactions on services computing*, 2017.

[4] S. Wang, T. Tuor, T. Salonidis, K. K. Leung, C. Makaya, T. He, and K. Chan, "Adaptive federated learning in resource constrained edge computing systems," *IEEE Journal on Selected Areas in Communications*, vol. 37, no. 6, pp. 1205–1221, 2019.

[5] 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 of Things Journal*, pp. 1–1, 2020.

[6] W. Shi, J. Cao, Q. Zhang, Y. Li, and L. Xu, "Edge computing: Vision and challenges," *IEEE internet of things journal*, vol. 3, no. 5, pp. 637–646, 2016.

[7] S. Arisdakessian, O. A. Wahab, A. Mourad, H. Otrok, and N. Kara, "FoGMatch: An Intelligent Multi-Criteria IoT-Fog Scheduling Approach Using Game Theory," *IEEE/ACM Transactions on Networking*, 2020.

[8] H. B. McMahan, E. Moore, D. Ramage, S. Hampson *et al.*, "Communication-efficient learning of deep networks from decentralized data," in *Proceedings of the 20 th International Conference on Artificial Intelligence and Statistics (AISTATS)*, 2017.

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

[10] Q. Yang, Y. Liu, T. Chen, and Y. Tong, "Federated machine learning: Concept and applications," *ACM Transactions on Intelligent Systems and Technology (TIST)*, vol. 10, no. 2, pp. 1–19, 2019.

[11] P. Kairouz, H. B. McMahan, B. Avent, A. Bellet, M. Bennis, A. N. Bhagoji, K. Bonawitz, Z. Charles, G. Cormode, R. Cummings *et al.*, "Advances and open problems in federated learning," *arXiv preprint arXiv:1912.04977*, 2019.

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

[13] T. Li, A. K. Sahu, A. Talwalkar, and V. Smith, "Federated learning: Challenges, methods, and future directions," *IEEE Signal Processing Magazine*, vol. 37, no. 3, pp. 50–60, 2020.

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

[15] W. Y. B. Lim, N. C. Luong, D. T. Hoang, Y. Jiao, Y.-C. Liang, Q. Yang, D. Niyato, and C. Miao, "Federated learning in mobile edge networks: A comprehensive survey," *IEEE Communications Surveys & Tutorials*, 2020.

[16] J. Xu and F. Wang, "Federated learning for healthcare informatics," *arXiv preprint arXiv:1911.06270*, 2019.

[17] R. Gu, S. Yang, and F. Wu, "Distributed machine learning on mobile devices: A survey," *arXiv preprint arXiv:1909.08329*, 2019.

[18] S. S. Saab and D. Shen, "Multidimensional gains for stochastic approximation," *IEEE Transactions on Neural Networks and Learning Systems*, vol. 31, no. 5, pp. 1602–1615, 2020.

[19] M. Orabi, J. Khalife, A. A. Abdallah, Z. M. Kassas, and S. S. Saab, "A machine learning approach for gps code phase estimation in multipath environments," in *2020 IEEE/ION Position, Location and Navigation Symposium (PLANS)*, 2020, pp. 1224–1229.

[20] X. Li, K. Huang, W. Yang, S. Wang, and Z. Zhang, "On the convergence of fedavg on non-iid data," in *Proceedings of the International Conference on Learning Representations*, 2019.

[21] T. Li, A. K. Sahu, M. Zaheer, M. Sanjabi, A. Talwalkar, and V. Smith, "Federated optimization in heterogeneous networks," *Proceedings of the Third Conference on Machine Learning and Systems (MLSys)*, 2020.

[22] A. Reisizadeh, A. Mokhtari, H. Hassani, A. Jadbabaie, and R. Pedarsani, "Fedpaq: A communication-efficient federated learning method with periodic averaging and quantization," in *International Conference on Artificial Intelligence and Statistics*. PMLR, 2020, pp. 2021–2031.

[23] J. So, B. Guler, and A. S. Avestimehr, "Turbo-aggregate: Breaking the quadratic aggregation barrier in secure federated learning," *arXiv preprint arXiv:2002.04156*, 2020.

[24] H. Wang, M. Yurochkin, Y. Sun, D. Papailiopoulos, and Y. Khazaeni, "Federated learning with matched averaging," in *International Conference on Learning Representations*, 2019.

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

[26] K. Bonawitz, H. Eichner, W. Grieskamp, D. Huba, A. Ingerman, V. Ivanov, C. Kiddon, J. Konečnỳ, S. Mazzocchi, H. B. McMahan *et al.*, "Towards federated learning at scale: System design," *arXiv preprint arXiv:1902.01046*, 2019.

[27] J. Dean and S. Ghemawat, "Mapreduce: simplified data processing on large clusters," *Communications of the ACM*, vol. 51, no. 1, pp. 107–113, 2008.

[28] B. Szorenyi, R. Busa-Fekete, I. Hegedus, R. Ormándi, M. Jelasity, and B. Kégl, "Gossip-based distributed stochastic bandit algorithms," in *International Conference on Machine Learning*, 2013, pp. 19–27.

[29] J. M. Rushby, "Design and verification of secure systems," *ACM SIGOPS Operating Systems Review*, vol. 15, no. 5, pp. 12–21, 1981.

[30] M. Sabt, M. Achemlal, and A. Bouabdallah, "Trusted execution environment: what it is, and what it is not," in *2015 IEEE Trustcom/BigDataSE/ISPA*, vol. 1. IEEE, 2015, pp. 57–64.

[31] J. A. Stankovic, "Research directions for the internet of things," *IEEE Internet of Things Journal*, vol. 1, no. 1, pp. 3–9, 2014.

[32] A. Chamra and H. Harmanani, "A smart green house control and management system using iot," in *17th International Conference on Information Technology–New Generations (ITNG 2020). Advances in Intelligent Systems and Computing*, vol. 1134, 2020.

[33] X. Wang, Y. Han, V. C. Leung, D. Niyato, X. Yan, and X. Chen, "Convergence of edge computing and deep learning: A comprehensive survey," *IEEE Communications Surveys & Tutorials*, vol. 22, no. 2, pp. 869–904, 2020.

[34] H. Li, K. Ota, and M. Dong, "Learning iot in edge: Deep learning for the internet of things with edge computing," *IEEE network*, vol. 32, no. 1, pp. 96–101, 2018.

[35] N. Abbas, Y. Zhang, A. Taherkordi, and T. Skeie, "Mobile edge computing: A survey," *IEEE Internet of Things Journal*, vol. 5, no. 1, pp. 450–465, 2017.

[36] T. Taleb, K. Samdanis, B. Mada, H. Flinck, S. Dutta, and D. Sabella, "On multi-access edge computing: A survey of the emerging 5g network edge cloud architecture and orchestration," *IEEE Communications Surveys & Tutorials*, vol. 19, no. 3, pp. 1657–1681, 2017.

[37] R. S. Sinha, Y. Wei, and S.-H. Hwang, "A survey on lpwa technology: Lora and nb-iot," *Ict Express*, vol. 3, no. 1, pp. 14–21, 2017.

[38] R. Shafin, L. Liu, V. Chandrasekhar, H. Chen, J. Reed, and J. C. Zhang, "Artificial intelligence-enabled cellular networks: A critical path to beyond-5g and 6g," *IEEE Wireless Communications*, vol. 27, no. 2, pp. 212–217, 2020.

[39] Y. Liu, X. Yuan, Z. Xiong, J. Kang, X. Wang, and D. Niyato, "Federated learning for 6g communications: Challenges, methods, and future directions," *arXiv preprint arXiv:2006.02931*, 2020.

[40] S. Samarakoon, M. Bennis, W. Saad, and M. Debbah, "Distributed federated learning for ultra-reliable low-latency vehicular communications," *IEEE Transactions on Communications*, 2019.

[41] S. Samarakoon, M. Bennis, W. Saad, and M. Debbah, "Federated learning for ultra-reliable low-latency v2v communications," in *Proceedings of the IEEE Global Communications Conference (GLOBECOM)*. IEEE, 2018, pp. 1–7.

[42] J. Ren, H. Wang, T. Hou, S. Zheng, and C. Tang, "Federated learning-based computation offloading optimization in edge computing-supported internet of things," *IEEE Access*, vol. 7, pp. 69 194–69 201, 2019.

[43] 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. 23 920–23 935, 2020.

[44] M. S. H. Abad, E. Ozfatura, D. Gunduz, and O. Ercetin, "Hierarchical federated learning across heterogeneous cellular networks," in *Proceedings of the IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE, 2020, pp. 8866–8870.

[45] F. Ang, L. Chen, N. Zhao, Y. Chen, W. Wang, and F. R. Yu, "Robust federated learning with noisy communication," *IEEE Transactions on Communications*, 2020.

[46] Y. Lu, X. Huang, K. Zhang, S. Maharjan, and Y. Zhang, "Blockchain empowered asynchronous federated learning for secure data sharing in internet of vehicles," *IEEE Transactions on Vehicular Technology*, vol. 69, no. 4, pp. 4298–4311, 2020.

[47] H. H. Yang, Z. Liu, T. Q. Quek, and H. V. Poor, "Scheduling policies for federated learning in wireless networks," *IEEE Transactions on Communications*, 2019.

[48] D. Ebrahimi, S. Sharafeddine, P. Ho, and C. Assi, "Autonomous uav trajectory for localizing ground objects: A reinforcement learning approach," *IEEE Transactions on Mobile Computing*, pp. 1–1, 2020.

[49] M. Samir, D. Ebrahimi, C. Assi, S. Sharafeddine, and A. Ghrayeb, "Trajectory planning of multiple dronecells in vehicular networks: A reinforcement learning approach," *IEEE Networking Letters*, vol. 2, no. 1, pp. 14–18, 2020.

[50] W. Y. B. Lim, J. Huang, Z. Xiong, J. Kang, D. Niyato, X.-S. Hua, C. Leung, and C. Miao, "Towards federated learning in uav-enabled internet of vehicles: A multi-dimensional contract-matching approach," *arXiv preprint arXiv:2004.03877*, 2020.

[51] H. Shiri, J. Park, and M. Bennis, "Communication-efficient massive uav online path control: Federated learning meets mean-field game theory," *arXiv preprint arXiv:2003.04451*, 2020.

[52] Y. Liu, J. Nie, X. Li, S. H. Ahmed, W. Y. B. Lim, and C. Miao, "Federated learning in the sky: Aerial-ground air quality sensing framework with uav swarms," *IEEE Internet of Things Journal*, 2020.

[53] 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," *arXiv preprint arXiv:2002.08196*, 2020.

[54] J. S. Ng, W. Y. B. Lim, H.-N. Dai, Z. Xiong, J. Huang, D. Niyato, X.-S. Hua, C. Leung, and C. Miao, "Joint auction-coalition formation framework for communication-efficient federated learning in uav-enabled internet of vehicles," *arXiv preprint arXiv:2007.06378*, 2020.

[55] M. Duan, D. Liu, X. Chen, Y. Tan, J. Ren, L. Qiao, and L. Liang, "Astraea: Self-balancing federated learning for improving classification accuracy of mobile deep learning applications," in *Proceedong of the IEEE 37th International Conference on Computer Design (ICCD)*. IEEE, 2019, pp. 246–254.

[56] Y. Zhao, M. Li, L. Lai, N. Suda, D. Civin, and V. Chandra, "Federated learning with non-iid data," *arXiv preprint arXiv:1806.00582*, 2018.

[57] N. Yoshida, T. Nishio, M. Morikura, K. Yamamoto, and R. Yonetani, "Hybrid-fl: Cooperative learning mechanism using non-iid data in wireless networks," in *Proceedings of the International Conference on Communications (ICC)*, 2020.

[58] E. Jeong, S. Oh, H. Kim, J. Park, M. Bennis, and S.-L. Kim, "Communication-efficient on-device machine learning: Federated distillation and augmentation under non-iid private data," in *2nd Workshop on Machine Learning on the Phone and other Consumer Devices (MLPCD 2) at NeurIPS*, 2018.

[59] J. Goetz, K. Malik, D. Bui, S. Moon, H. Liu, and A. Kumar, "Active federated learning," *arXiv preprint arXiv:1909.12641*, 2019.

[60] V. Smith, C.-K. Chiang, M. Sanjabi, and A. S. Talwalkar, "Federated multi-task learning," in *Advances in Neural Information Processing Systems*, 2017, pp. 4424–4434.

[61] S. Caldas, V. Smith, and A. Talwalkar, "Federated kernelized multi-task learning," in *Proceeding of the SysML Conference*, 2018.

[62] F. Sattler, K.-R. Müller, and W. Samek, "Clustered federated learning: Model-agnostic distributed multi-task optimization under privacy constraints," *arXiv preprint arXiv:1910.01991*, 2019.

[63] A. Ghosh, J. Hong, D. Yin, and K. Ramchandran, "Robust federated learning in a heterogeneous environment," *arXiv preprint arXiv:1906.06629*, 2019.

[64] C. Briggs, Z. Fan, and P. Andras, "Federated learning with hierarchical clustering of local updates to improve training on non-iid data," *arXiv preprint arXiv:2004.11791*, 2020.

[65] Y. Liu, T. Chen, and Q. Yang, "Secure federated transfer learning," *arXiv preprint arXiv:1812.03337*, 2018.

[66] S. Sharma, X. Chaoping, Y. Liu, and Y. Kang, "Secure and efficient federated transfer learning," *IEEE Big Data*, 2019.

[67] W. Liu, L. Chen, Y. Chen, and W. Zhang, "Accelerating federated learning via momentum gradient descent," *IEEE Transactions on Parallel and Distributed Systems*, vol. 31, no. 8, pp. 1754–1766, 2020.

[68] Z. Jiang, A. Balu, C. Hegde, and S. Sarkar, "Collaborative deep learning in fixed topology networks," in *Advances in Neural Information Processing Systems*, 2017, pp. 5904–5914.

[69] A. Koskela and A. Honkela, "Learning rate adaptation for federated and differentially private learning," *arXiv preprint arXiv:1809.03832*, 2018.

[70] X. Chen, T. Chen, H. Sun, Z. S. Wu, and M. Hong, "Distributed training with heterogeneous data: Bridging median and mean based algorithms," *arXiv preprint arXiv:1906.01736*, 2019.

[71] H. Mostafa, "Robust federated learning through representation matching and adaptive hyper-parameters," in *International Conference on Learning Representations*, 2020.

[72] Y. Ruan, X. Zhang, S.-C. Liang, and C. Joe-Wong, "Towards flexible device participation in federated learning for non-iid data," *arXiv preprint arXiv:2006.06954*, 2020.

[73] H. Wang, Z. Kaplan, D. Niu, and B. Li, "Optimizing federated learning on non-iid data with reinforcement learning," in *IEEE INFOCOM 2020-IEEE Conference on Computer Communications*. IEEE, 2020, pp. 1698–1707.

[74] H. Eichner, T. Koren, B. Mcmahan, N. Srebro, and K. Talwar, "Semi-cyclic stochastic gradient descent," in *Proceedings of the 36th International Conference on Machine Learning*, vol. 97, 2019.

[75] D. Li and J. Wang, "Fedmd: Heterogenous federated learning via model distillation," in *NeurIPS 2019 Workshop on Federated Learning for Data Privacy and Confidentiality*, 2019.

[76] M. Mohri, G. Sivek, and A. T. Suresh, "Agnostic federated learning," *International Conference on Machine Learning (ICML)*, 2019.

[77] T. Li, M. Sanjabi, A. Beirami, and V. Smith, "Fair resource allocation in federated learning," in *International Conference on Learning Representations*, 2019.

[78] D. C. Verma, G. White, S. Julier, S. Pasteris, S. Chakraborty, and G. Cirincione, "Approaches to address the data skew problem in federated learning," in *Artificial Intelligence and Machine Learning for Multi-Domain Operations Applications*, vol. 11006. International Society for Optics and Photonics, 2019, p. 110061I.

[79] Y. Bengio, G. Mesnil, Y. Dauphin, and S. Rifai, "Better mixing via deep representations," in *International conference on machine learning*, 2013, pp. 552–560.

[80] F. Haddadpour and M. Mahdavi, "On the convergence of local descent methods in federated learning," in *Proceedings of the International Conference on Learning Representations*, 2020.

[81] Q. Jing, W. Wang, J. Zhang, H. Tian, and K. Chen, "Quantifying the performance of federated transfer learning," *arXiv preprint arXiv:1912.12795*, 2019.

[82] K. Cheng, T. Fan, Y. Jin, Y. Liu, T. Chen, and Q. Yang, "Secureboost: A lossless federated learning framework," *arXiv preprint arXiv:1901.08755*, 2019.

[83] C. J. Watkins and P. Dayan, "Q-learning," *Machine learning*, vol. 8, no. 3-4, pp. 279–292, 1992.

[84] C.-W. Tsai, C.-F. Lai, M.-C. Chiang, and L. T. Yang, "Data mining for internet of things: A survey," *IEEE Communications Surveys & Tutorials*, vol. 16, no. 1, pp. 77–97, 2013.

[85] A. L. Buczak and E. Guven, "A survey of data mining and machine learning methods for cyber security intrusion detection," *IEEE Communications surveys & tutorials*, vol. 18, no. 2, pp. 1153–1176, 2015.

[86] D. Xu, S. Yuan, L. Zhang, and X. Wu, "Fairgan: Fairness-aware generative adversarial networks," in *Proceedings of the IEEE International Conference on Big Data (Big Data)*. IEEE, 2018, pp. 570–575.

[87] B. H. Zhang, B. Lemoine, and M. Mitchell, "Mitigating unwanted biases with adversarial learning," in *Proceedings of the AAAI/ACM Conference on AI, Ethics, and Society*, 2018, pp. 335–340.

[88] N. Agarwal, A. T. Suresh, F. X. X. Yu, S. Kumar, and B. McMahan, "cpsgd: Communication-efficient and differentially-private distributed sgd," in *Advances in Neural Information Processing Systems*, 2018, pp. 7564–7575.

[89] F. Sattler, S. Wiedemann, K.-R. Müller, and W. Samek, "Robust and communication-efficient federated learning from non-iid data," *IEEE transactions on neural networks and learning systems*, 2019.

[90] M. M. Amiri and D. Gunduz, "Federated learning over wireless fading channels," *IEEE Transactions on Wireless Communications*, 2019.

[91] S. Caldas, J. Konečny, H. B. McMahan, and A. Talwalkar, "Expanding the reach of federated learning by reducing client resource requirements," *arXiv preprint arXiv:1812.07210*, 2018.

[92] M. Yurochkin, M. Agarwal, S. Ghosh, K. Greenewald, N. Hoang, and Y. Khazaeni, "Bayesian nonparametric federated learning of neural networks," in *International Conference on Machine Learning*, 2019, pp. 7252–7261.

[93] H. Zhu and Y. Jin, "Multi-objective evolutionary federated learning," *IEEE transactions on neural networks and learning systems*, 2019.

[94] C. Niu, F. Wu, S. Tang, L. Hua, R. Jia, C. Lv, Z. Wu, and G. Chen, "Secure federated submodel learning," *arXiv preprint arXiv:1911.02254*, 2019.

[95] N. Guha, A. Talwlkar, and V. Smith, "One-shot federated learning," in *2nd Workshop on Machine Learning on the Phone and other Consumer Devices at NeurIPs*, 2018.

[96] Y. Liu, Y. Kang, X. Zhang, L. Li, Y. Cheng, T. Chen, M. Hong, and Q. Yang, "A communication efficient vertical federated learning framework," *arXiv preprint arXiv:1912.11187*, 2019.

[97] M. Kamp, L. Adilova, J. Sicking, F. Hüger, P. Schlicht, T. Wirtz, and S. Wrobel, "Efficient decentralized deep learning by dynamic model averaging," in *Proceedings of the Joint European Conference on Machine Learning and Knowledge Discovery in Databases*. Springer, 2018, pp. 393–409.

[98] 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 Network*, vol. 33, no. 5, pp. 156–165, 2019.

[99] S. Feng, D. Niyato, P. Wang, D. I. Kim, and Y.-C. Liang, "Joint service pricing and cooperative relay communication for federated learning," in *Proceedings of the International Conference on Internet of Things*

(iThings) and IEEE Green Computing and Communications (GreenCom) and IEEE Cyber, Physical and Social Computing (CPSCom) and IEEE Smart Data (SmartData). IEEE, 2019, pp. 815–820.

[100] Y. Sun, S. Zhou, and D. Gündüz, "Energy-aware analog aggregation for federated learning with redundant data," *IEEE International Conference on Communications (ICC)*, 2020.

[101] K. Yang, T. Jiang, Y. Shi, and Z. Ding, "Federated learning via over-the-air computation," *IEEE Transactions on Wireless Communications*, 2020.

[102] M. M. Amiri and D. Gündüz, "Machine learning at the wireless edge: Distributed stochastic gradient descent over-the-air," *IEEE Transactions on Signal Processing*, vol. 68, pp. 2155–2169, 2020.

[103] W. Wu, L. He, W. Lin, R. Mao, C. Maple, and S. A. Jarvis, "Safa: a semi-asynchronous protocol for fast federated learning with low overhead," *IEEE Transactions on Computers*, 2020.

[104] Y. Chen, Y. Ning, and H. Rangwala, "Asynchronous online federated learning for edge devices," *arXiv preprint arXiv:1911.02134*, 2019.

[105] P. Pinyoanuntapong, P. Janakaraj, P. Wang, M. Lee, and C. Chen, "Fedair: Towards multi-hop federated learning over-the-air," in *2020 IEEE 21st International Workshop on Signal Processing Advances in Wireless Communications (SPAWC)*. IEEE, 2020, pp. 1–5.

[106] Y.-D. Lin, W.-M. Yin, and C.-Y. Huang, "An investigation into hfc mac protocols: mechanisms, implementation, and research issues," *IEEE Communications Surveys & Tutorials*, vol. 3, no. 3, pp. 2–13, 2000.

[107] J. Liu, N. Kato, J. Ma, and N. Kadowaki, "Device-to-device communication in lte-advanced networks: A survey," *IEEE Communications Surveys & Tutorials*, vol. 17, no. 4, pp. 1923–1940, 2014.

[108] M. P. McGarry, M. Reisslein, and M. Maier, "Ethernet passive optical network architectures and dynamic bandwidth allocation algorithms," *IEEE Communications Surveys & Tutorials*, vol. 10, no. 3, pp. 46–60, 2008.

[109] Z. Li, H. Huang, and S. Misra, "Compressed sensing via dictionary learning and approximate message passing for multimedia internet of things," *IEEE Internet of Things Journal*, vol. 4, no. 2, pp. 505–512, 2016.

[110] S. L. Keoh, S. S. Kumar, and H. Tschofenig, "Securing the internet of things: A standardization perspective," *IEEE Internet of things Journal*, vol. 1, no. 3, pp. 265–275, 2014.

[111] N. Nguyen-Thanh, P. Ciblat, S. Maleki, and V.-T. Nguyen, "How many bits should be reported in quantized cooperative spectrum sensing?" *IEEE Wireless Communications Letters*, vol. 4, no. 5, pp. 465–468, 2015.

[112] M. B. Khalilsarai, S. Haghighatshoar, X. Yi, and G. Caire, "Fdd massive mimo via ul/dl channel covariance extrapolation and active channel sparsification," *IEEE Transactions on Wireless Communications*, vol. 18, no. 1, pp. 121–135, 2018.

[113] H. Ji, B. Shim, and J. W. Choi, "Channel sparsification beamforming for internet-of-things systems," in *2017 IEEE International Conference on Communications (ICC)*. IEEE, 2017, pp. 1–5.

[114] G. Zhu and K. Huang, "Mimo over-the-air computation for high-mobility multimodal sensing," *IEEE Internet of Things Journal*, vol. 6, no. 4, pp. 6089–6103, 2018.

[115] C. R. Lin and M. Gerla, "Asynchronous multimedia multihop wireless networks," in *Proceedings of INFOCOM'97*, vol. 1. IEEE, 1997, pp. 118–125.

[116] B. Liu, B. Rong, R. Q. Hu, and Y. Qian, "Neighbor discovery algorithms in directional antenna based synchronous and asynchronous wireless ad hoc networks," *IEEE wireless communications*, vol. 20, no. 6, pp. 106–112, 2013.

[117] P. P. Liang, T. Liu, L. Ziyin, R. Salakhutdinov, and L.-P. Morency, "Think locally, act globally: Federated learning with local and global representations," in *Workshop on Federated Learning at NeurIPS*, 2019.

[118] 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," *arXiv preprint arXiv:1909.07972*, 2019.

[119] C. Hu, J. Jiang, and Z. Wang, "Decentralized federated learning: A segmented gossip approach," in *1st International Workshop on Federated Machine Learning for User Privacy and Data Confidentiality (FML'19)*, 2019.

[120] Z. Zhou, S. Yang, L. J. Pu, and S. Yu, "Cefl: Online admission control, data scheduling and accuracy tuning for cost-efficient federated learning across edge nodes," *IEEE Internet of Things Journal*, 2020.

[121] H. T. Nguyen, N. C. Luong, J. Zhao, C. Yuen, and D. Niyato, "Resource allocation in mobility-aware federated learning networks: A deep reinforcement learning approach," *arXiv preprint arXiv:1910.09172*, 2019.

[122] T. T. Anh, N. C. Luong, D. Niyato, D. I. Kim, and L.-C. Wang, "Efficient training management for mobile crowd-machine learning: A deep reinforcement learning approach," *IEEE Wireless Communications Letters*, vol. 8, no. 5, pp. 1345–1348, 2019.

[123] T. Nishio and R. Yonetani, "Client selection for federated learning with heterogeneous resources in mobile edge," in *Proceedings of the IEEE International Conference on Communications (ICC)*. IEEE, 2019, pp. 1–7.

[124] 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 of Things Journal*, vol. 6, no. 6, pp. 10 700–10 714, 2019.

[125] J. Kang, Z. Xiong, D. Niyato, Y. Zou, Y. Zhang, and M. Guizani, "Reliable federated learning for mobile networks," *IEEE Wireless Communications*, 2020.

[126] C. He, C. Tan, H. Tang, S. Qiu, and J. Liu, "Central server free federated learning over single-sided trust social networks," *arXiv preprint arXiv:1910.04956*, 2019.

[127] G. Wang, C. X. Dang, and Z. Zhou, "Measure contribution of participants in federated learning," in *2019 IEEE International Conference on Big Data (Big Data)*. IEEE, 2019, pp. 2597–2604.

[128] S. A. Rahman, H. Tout, A. Mourad, and C. Talhi, "Fedmccs: Multi criteria client selection model for optimal iot federated learning," *IEEE Internet of Things Journal*, pp. 1–1, 2020.

[129] X. Yuan, "Heuristic algorithms for multiconstrained quality-of-service routing," *IEEE/ACM transactions on networking*, vol. 10, no. 2, pp. 244–256, 2002.

[130] H. Lin and H. Üster, "Exact and heuristic algorithms for data-gathering cluster-based wireless sensor network design problem," *IEEE/ACM transactions on networking*, vol. 22, no. 3, pp. 903–916, 2013.

[131] H. W. Kuhn, "The hungarian method for the assignment problem," *Naval research logistics quarterly*, vol. 2, no. 1-2, pp. 83–97, 1955.

[132] M. A. Salahuddin, A. Al-Fuqaha, and M. Guizani, "Reinforcement learning for resource provisioning in the vehicular cloud," *IEEE Wireless Communications*, vol. 23, no. 4, pp. 128–135, 2016.

[133] M. Mohammadi, A. Al-Fuqaha, M. Guizani, and J.-S. Oh, "Semisupervised deep reinforcement learning in support of iot and smart city services," *IEEE Internet of Things Journal*, vol. 5, no. 2, pp. 624–635, 2017.

[134] L. Cui, C. Xu, S. Yang, J. Z. Huang, J. Li, X. Wang, Z. Ming, and N. Lu, "Joint optimization of energy consumption and latency in mobile edge computing for internet of things," *IEEE Internet of Things Journal*, vol. 6, no. 3, pp. 4791–4803, 2018.

[135] J. Mills, J. Hu, and G. Min, "Communication-efficient federated learning for wireless edge intelligence in iot," *IEEE Internet of Things Journal*, 2019.

[136] R. T. Rockafellar, "Lagrange multipliers and optimality," *SIAM review*, vol. 35, no. 2, pp. 183–238, 1993.

[137] O. A. Wahab, J. Bentahar, H. Otrok, and A. Mourad, "Towards trustworthy multi-cloud services communities: A trust-based hedonic coalitional game," *IEEE Transactions on Services Computing*, vol. 11, no. 1, pp. 184–201, 2016.

[138] O. A. Wahab, J. Bentahar, H. Otrok, and A. Mourad, "A survey on trust and reputation models for web services: Single, composite, and communities," *Decision Support Systems*, vol. 74, pp. 121–134, 2015.

[139] G. Rjoub, O. A. Wahab, J. Bentahar, and A. Bataineh, "A trust and energy-aware double deep reinforcement learning scheduling strategy for federated learning on iot devices," in *International Conference on Service-Oriented Computing*. Springer, 2020, pp. 319–333.

[140] A. E. Roth, *The Shapley value: essays in honor of Lloyd S. Shapley*. Cambridge University Press, 1988.

[141] O. A. Wahab, J. Bentahar, H. Otrok, and A. Mourad, "I know you are watching me: Stackelberg-based adaptive intrusion detection strategy for insider attacks in the cloud," in *2017 IEEE International Conference on Web Services (ICWS)*. IEEE, 2017, pp. 728–735.

[142] O. A. Wahab, J. Bentahar, H. Otrok, and A. Mourad, "A stackelberg game for distributed formation of business-driven services communities," *Expert Systems with Applications*, vol. 45, pp. 359–372, 2016.

[143] S. Maharjan, Q. Zhu, Y. Zhang, S. Gjessing, and T. Basar, "Dependable demand response management in the smart grid: A stackelberg game approach," *IEEE Transactions on Smart Grid*, vol. 4, no. 1, pp. 120–132, 2013.

[144] O. A. Wahab, R. Cohen, J. Bentahar, H. Otrok, A. Mourad, and G. Rjoub, "An endorsement-based trust bootstrapping approach for newcomer cloud services," *Information Sciences*, vol. 527, pp. 159–175, 2020.

[145] P. Blanchard, R. Guerraoui, J. Stainer *et al.*, "Machine learning with adversaries: Byzantine tolerant gradient descent," in *Advances in Neural Information Processing Systems*, 2017, pp. 119–129.

[146] M. Fang, X. Cao, J. Jia, and N. Z. Gong, "Local model poisoning attacks to byzantine-robust federated learning," in *Usenix Security Symposium*, 2020.

[147] L. Muñoz-González, K. T. Co, and E. C. Lupu, "Byzantine-robust federated machine learning through adaptive model averaging," *arXiv preprint arXiv:1909.05125*, 2019.

[148] L. Li, W. Xu, T. Chen, G. B. Giannakis, and Q. Ling, "Rsa: Byzantine-robust stochastic aggregation methods for distributed learning from heterogeneous datasets," in *Proceedings of the AAAI Conference on Artificial Intelligence*, vol. 33, 2019, pp. 1544–1551.

[149] S. Li, Y. Cheng, Y. Liu, W. Wang, and T. Chen, "Abnormal client behavior detection in federated learning," in *2nd International Workshop on Federated Learning for Data Privacy and Confidentiality, in Conjunction with NeurIPS 2019 (FL-NeurIPS 19)*, 2019.

[150] A. N. Bhagoji, S. Chakraborty, P. Mittal, and S. Calo, "Analyzing federated learning through an adversarial lens," in *International Conference on Machine Learning (ICML)*, vol. 97, 2019, pp. 634–643.

[151] C. Fung, C. J. Yoon, and I. Beschastnikh, "Mitigating sybils in federated learning poisoning," *arXiv preprint arXiv:1808.04866*, 2018.

[152] S. Li, Y. Cheng, W. Wang, Y. Liu, and T. Chen, "Learning to detect malicious clients for robust federated learning," *arXiv preprint arXiv:2002.00211*, 2020.

[153] S. A. Rahman, H. Tout, C. Talhi, and A. Mourad, "Internet of things intrusion detection: Centralized, on-device, or federated learning?" *IEEE Network*, vol. 34, no. 6, pp. 310–317, 2020.

[154] E. Bagdasaryan, A. Veit, Y. Hua, D. Estrin, and V. Shmatikov, "How to backdoor federated learning," *arXiv preprint arXiv:1807.00459*, 2018.

[155] T. Yu, X. Wang, and A. Shami, "Recursive principal component analysis-based data outlier detection and sensor data aggregation in iot systems," *IEEE Internet of Things Journal*, vol. 4, no. 6, pp. 2207–2216, 2017.

[156] Z. Li, Z. Huang, C. Chen, and C. Hong, "Quantification of the leakage in federated learning," in *Workshop on Federated Learning for Data Privacy and Confidentiality in Conjunction with NeurIPS 2019*, 2019.

[157] L. Melis, C. Song, E. De Cristofaro, and V. Shmatikov, "Exploiting unintended feature leakage in collaborative learning," in *Proceedings of the IEEE Symposium on Security and Privacy (SP)*. IEEE, 2019, pp. 691–706.

[158] R. C. Geyer, T. Klein, and M. Nabi, "Differentially private federated learning: A client level perspective," in *NIPS Workshop: Machine Learning on the Phone and other Consumer Devices*, 2017.

[159] D. Peterson, P. Kanani, and V. J. Marathe, "Private federated learning with domain adaptation," in *Workshop on Federated Learning for Data Privacy and Confidentiality (in Conjunction with NeurIPS 2019)*, 2019.

[160] Z. Liu, T. Li, V. Smith, and V. Sekar, "Enhancing the privacy of federated learning with sketching," *arXiv preprint arXiv:1911.01812*, 2019.

[161] A. Triastcyn and B. Faltings, "Federated generative privacy," *IEEE Intelligent Systems*, 2020.

[162] Y. Han and X. Zhang, "Robust federated training via collaborative machine teaching using trusted instances," *arXiv preprint arXiv:1905.02941*, 2019.

[163] S. Hardy, W. Henecka, H. Ivey-Law, R. Nock, G. Patrini, G. Smith, and B. Thorne, "Private federated learning on vertically partitioned data via entity resolution and additively homomorphic encryption," *arXiv preprint arXiv:1711.10677*, 2017.

[164] K. Mandal and G. Gong, "Privfl: Practical privacy-preserving federated regressions on high-dimensional data over mobile networks," in *Proceedings of the 2019 ACM SIGSAC Conference on Cloud Computing Security Workshop*, 2019, pp. 57–68.

[165] Y. Feng, X. Yang, W. Fang, S.-T. Xia, and X. Tang, "Practical and bilateral privacy-preserving federated learning," *arXiv preprint arXiv:2002.09843*, 2020.

[166] K. Bonawitz, V. Ivanov, B. Kreuter, A. Marcedone, H. B. McMahan, S. Patel, D. Ramage, A. Segal, and K. Seth, "Practical secure aggregation for privacy-preserving machine learning," in *Proceedings of the 2017 ACM SIGSAC Conference on Computer and Communications Security*, 2017, pp. 1175–1191.

[167] K. Bonawitz, V. Ivanov, B. Kreuter, A. Marcedone, H. B. McMahan, S. Patel, D. Ramage, A. Segal, and K. Seth, "Practical secure aggregation for federated learning on user-held data," in *NIPS Workshop on Private Multi-Party Machine Learning*, 2016.

[168] S. Truex, N. Baracaldo, A. Anwar, T. Steinke, H. Ludwig, R. Zhang, and Y. Zhou, "A hybrid approach to privacy-preserving federated learning," in *Proceedings of the 12th ACM Workshop on Artificial Intelligence and Security*, 2019, pp. 1–11.

[169] H. Kim, J. Park, M. Bennis, and S.-L. Kim, "Blockchained on-device federated learning," *IEEE Communications Letters*, 2019.

[170] C. Ma, J. Li, M. Ding, H. H. Yang, F. Shu, T. Q. Quek, and H. V. Poor, "On safeguarding privacy and security in the framework of federated learning," *IEEE Network*, 2020.

[171] J. Liu, C. Zhang, and Y. Fang, "Epic: A differential privacy framework to defend smart homes against internet traffic analysis," *IEEE Internet of Things Journal*, vol. 5, no. 2, pp. 1206–1217, 2018.

[172] M. Du, K. Wang, Y. Chen, X. Wang, and Y. Sun, "Big data privacy preserving in multi-access edge computing for heterogeneous internet of things," *IEEE Communications Magazine*, vol. 56, no. 8, pp. 62–67, 2018.

[173] A. Dorri, M. Steger, S. S. Kanhere, and R. Jurdak, "Blockchain: A distributed solution to automotive security and privacy," *IEEE Communications Magazine*, vol. 55, no. 12, pp. 119–125, 2017.

[174] F. Tschorsch and B. Scheuermann, "Bitcoin and beyond: A technical survey on decentralized digital currencies," *IEEE Communications Surveys & Tutorials*, vol. 18, no. 3, pp. 2084–2123, 2016.

[175] Y. Yu, Y. Li, J. Tian, and J. Liu, "Blockchain-based solutions to security and privacy issues in the internet of things," *IEEE Wireless Communications*, vol. 25, no. 6, pp. 12–18, 2018.

[176] Y. Sarikaya and O. Ercetin, "Motivating workers in federated learning: A stackelberg game perspective," *IEEE Networking Letters*, 2019.

[177] T. Song, Y. Tong, and S. Wei, "Profit allocation for federated learning," in *Proceedings of the IEEE International Conference on Big Data (Big Data)*. IEEE, 2019, pp. 2577–2586.

[178] Y. Zhan, P. Li, Z. Qu, D. Zeng, and S. Guo, "A learning-based incentive mechanism for federated learning," *IEEE Internet of Things Journal*, 2020.

[179] 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 Transactions on Wireless Communications*, 2020.

[180] Y. Jiao, P. Wang, D. Niyato, B. Lin, and D. I. Kim, "Toward an automated auction framework for wireless federated learning services market," *IEEE Transactions on Mobile Computing*, 2020.

[181] J. Kang, Z. Xiong, D. Niyato, H. Yu, Y.-C. Liang, and D. I. Kim, "Incentive design for efficient federated learning in mobile networks: A contract theory approach," in *Proceedings of the IEEE VTS Asia Pacific Wireless Communications Symposium (APWCS)*. IEEE, 2019, pp. 1–5.

[182] R. Trestian, O. Ormond, and G.-M. Muntean, "Game theory-based network selection: Solutions and challenges," *IEEE Communications surveys & tutorials*, vol. 14, no. 4, pp. 1212–1231, 2012.

[183] Y. Zhang, M. Pan, L. Song, Z. Dawy, and Z. Han, "A survey of contract theory-based incentive mechanism design in wireless networks," *IEEE wireless communications*, vol. 24, no. 3, pp. 80–85, 2017.

Prof. Omar Abdel Wahab is an assistant professor at the Department of Computer Science and Engineering, Université du Québec en Outaouais, Canada. He holds a Ph.D. in Information and Systems Engineering from Concordia University, Montreal, Canada. He received his Ms.c. in computer science in 2013 from the Lebanese American University (LAU), Lebanon. From 2017 to 2018, he was a postdoctoral fellow at the École de Technologie Supérieure (Canada), where he worked on an industrial research project in collaboration with Rogers and Ericsson. The main topics of his current research activities are in the areas of Internet of Things, artificial intelligence, cybersecurity and big data analytics. He is recipient of many prestigious grants from prestigious agencies in Canada such as the Natural Sciences and Engineering Research Council of Canada (NSERC) and Mitacs. Dr. Omar Abdel Wahab' research works in recent years have been published in many top-notch journals and conferences. He is a TPC member and publicity chair of several prestigious conferences and reviewer of several highly ranked journals.

Prof. Azzam Mourad received his M.Sc. in CS from Laval University, Canada (2003) and Ph.D. in ECE from Concordia University, Canada (2008). He is currently an associate professor of computer science with the Lebanese American University and an affiliate associate professor with the Software Engineering and IT Department, Ecole de Technologie Superieure (ETS), Montreal, Canada. He published more than 100 papers in international journal and conferences on Security, Network and Service Optimization and Management targeting IoT, Cloud/Fog/Edge Computing, Vehicular and Mobile Networks, and Federated Learning. He has served/serves as an associate editor for IEEE Transaction on Network and Service Management, IEEE Network, IEEE Open Journal of the Communications Society, IET Quantum Communication, and IEEE Communications Letters, the General Chair of IWCMC2020, the General Co-Chair of WiMob2016, and the Track Chair, a TPC member, and a reviewer for several prestigious journals and conferences. He is an IEEE senior member.

Prof. Hadi Otrok holds an associate professor position in the department of Electrical Engineering and Computer Science (EECS) at Khalifa University of Science and Technology, an affiliate associate professor in the Concordia Institute for Information Systems Engineering at Concordia University, Montreal, Canada, and an affiliate associate professor in the electrical department at Ecole de Technologie Superieure (ETS), Montreal, Canada. He received his Ph.D. in ECE from Concordia University. He is a senior member at IEEE, and associate editor at: IEEE TNSM, Ad-hoc networks (Elsevier), and IEEE Network. He served as associate editor at IEEE communications letters. He co-chaired several committees at various IEEE conferences. His research interests include: Blockchain, reinforcement learning, Federated Learning, crowd sensing and sourcing, ad hoc networks, and cloud and fog security.

Prof. Tarik Taleb is Professor at the School of Electrical Engineering, Aalto University, Finland. He is the founder and director of the MOSA!C Lab (www.mosaic-lab.org). He is also Professor at the Center of Wireless Communications, University of Oulu, Finland. Prior to his current positions, he was working as Senior Researcher and 3GPP Standards Expert at NEC Europe Ltd, Heidelberg, Germany. He was then leading the NEC Europe Labs Team working on R&D projects on carrier cloud platforms. Before joining NEC and till Mar. 2009, he worked as assistant professor at the Graduate School of Information Sciences, Tohoku University, Japan, in a lab fully funded by KDDI, the second largest mobile operator in Japan. From Oct. 2005 till Mar. 2006, he worked as research fellow at the Intelligent Cosmos Research Institute, Sendai, Japan. He received his B. E degree in Information Engineering with distinction, M.Sc. and Ph.D. degrees in Information Sciences from Tohoku Univ., in 2001, 2003, and 2005, respectively.

Prof. Taleb's research interests lie in the field of architectural enhancements to mobile core networks (particularly 3GPP's), network softwarization & slicing, mobile cloud networking, network function virtualization, software defined networking, mobile multimedia streaming, inter-vehicular communications, and cloud computing. Prof. Taleb has been also directly engaged in the development and standardization of the Evolved Packet System as a member of 3GPP's System Architecture working group 2. Prof. Taleb is a member of the IEEE Communications Society Standardization Program Development Board. As an attempt to bridge the gap between academia and industry, Prof. Taleb founded the "IEEE Workshop on Telecommunications Standards: from Research to Standards", a successful event that got awarded "best workshop award" by IEEE Communication Society (ComSoC). Based on the success of this workshop, Prof. Taleb has also founded and has been the steering committee chair of the IEEE Conf. on Standards for Communications and Networking.

Prof. Taleb served as the general chair of the 2019 edition of the IEEE Wireless Communications and Networking Conference (WCNC' 19) held in Marrakech, Morocco. He was the guest editor in chief of the IEEE JSAC Series on Network Softwarization & Enablers. He is/was on the editorial board of the IEEE Transactions on Wireless Communications, IEEE Wireless Communications Magazine, IEEE Journal on Internet of Things, IEEE Transactions on Vehicular Technology, IEEE Communications Surveys & Tutorials, and a number of Wiley journals. Till Dec. 2016, he served as chair of the Wireless Communications Technical Committee, the largest in IEEE ComSoC. He also served as Vice Chair of the Satellite and Space Communications Technical Committee of IEEE ComSoc (2006 - 2010).

Prof. Taleb is the recipient of the 2017 IEEE ComSoc Communications Software Technical Achievement Award (Dec. 2017) for his outstanding contributions to network softwarization. He is also the (co-) recipient of the 2017 IEEE Communications Society Fred W. Ellersick Prize (May 2017), the 2009 IEEE ComSoc Asia-Pacific Best Young Researcher award (Jun. 2009), the 2008 TELECOM System Technology Award from the Telecommunications Advancement Foundation (Mar. 2008), the 2007 Funai Foundation Science Promotion Award (Apr. 2007), the 2006 IEEE Computer Society Japan Chapter Young Author Award (Dec. 2006), the Niwa Yasujirou Memorial Award (Feb. 2005), and the Young Researcher's Encouragement Award from the Japan chapter of the IEEE Vehicular Technology Society (VTS) (Oct. 2003). Some of Prof. Taleb's research work have been also awarded best paper awards at prestigious IEEE-flagged conferences.