Though these smart devices, collectively, possess a large amount of valuable data. However, usually, the nature of this data is highly sensitive. Therefore, due to different constraints including privacy concerns and privacy laws like GDPR [12], China's cybersecurity law [13], and California's privacy right act [14], it has become almost impossible for companies to collect, transfer, use or integrate users' data without their consent for any specific purpose.

Traditionally in distributed learning environment, to train a model, we typically accumulate all data at a central location, properly distribute it to separate parties for processing. But now, due to more privacy concerns of people and extremely strict privacy laws, it is almost impossible to collect updated real-time users' private data at the central location.

In such scenarios, it is intuitive to smartly leverage the private data of users stored locally and perform the necessary computation (model training) on these devices. Thus, ensuring the privacy guarantee of users' personal data and, on the other hand, also utilizing the computational resources of client devices. Different collaborative learning techniques [15-17] have been proposed to train deep learning models where different clients collaborate with each other to update their models by leveraging the learned knowledge of other clients rather than their private data. Specifically, a very promising decentralized learning technique called federated learning [16, 17] has been coined which has instantly attracted a large research community in Machine learning towards this research direction i.e. rather than transferring data to code (computing), we move the code to data.

FL has many advantages as compared to traditional distributed machine learning approaches [18-21] like *privacy*, where devices don't have to share their private data with other devices including a centralized server. *Low latency*, as devices would have updated model locally, so they do not need to wait for inferencing from cloud-server. *Huge computational resources*, as usually hundreds of devices, could participate in FL so a lot of computational resources would be available to train the model. Similarly, FL can help to more efficiently utilize the network bandwidth as, now, there is no need to transmit raw data to cloud-server rather just need to share the trained model parameters.

Though FL has emerged as a remarkably effective decentralized learning framework to leverage the massively distributed, highly unbalanced, and Non-independent and Identical Distribution (IID) private data of smart devices. Nevertheless, it comes with many (unique) challenges related to data, model architecture, communication, and privacy. Like, here, data is typically expected to be massively distributed, Non-IID, unbalanced, and inaccessible by other devices or centralized server due to privacy constraints. Similarly, Communication cost could be much higher as compared to computation cost and could experience challenges of limited and inconsistent bandwidth for various devices and of passive sampling. Furthermore, participating devices may naturally require specialized or more personalized models based on their specific requirements. Likewise, privacy is one of the primary foci of decentralized learning so local data of devices would be inaccessible to any other party. Key challenges of federated learning have been addressed thoroughly in section 2.

Extensive works [22-34] have been performed which more or less covered different aspects of FL effectively. However, most of them usually discussed FL in some particular context or discussed the core challenges of FL in a limited way like Yang et al. [22] put their major focus on different categories of FL based on a different distribution of data. Kulkarni et al. [23] put their primary focus on model personalization techniques for FL but do not discuss the other issues of FL. Xia [24] draws a comparison of FL with deep learning while putting the main focus on applying watermarking on deep neural networks in FL. Lyu et al. [25] put their main focus of discussion to the potential threats to FL. Similarly, Li et al. [26] provide a comprehensive survey on FL systems but they mostly discuss the design aspects of FL. Aledhari et al. [32] present a comprehensive survey of FL while focusing on protocols, applications, and use cases of FL in detail. Another comprehensive survey is performed by Li et al [33] but they do not adequately address the personalization issues where local clients may contain diverse model architectures. Similarly, these studies [25, 34] effectively cover the privacy and security aspects of FL but do not adequately cover the other challenges of FL. Furthermore, many authors [27-32] have explained FL in a particular context or some potential solutions in that particular context.

To put it concisely, there are many surveys on FL. However, most of them are tutorial-based or comprehensive in a particular context. Thus it stimulates us to perform a concise and comprehensive survey on FL adequately covering its key issues with possible solutions and future research directions.

The rest of the paper is divided into the following sections. Sections 1.1, 1.2, 1.3, and 1.4 present the basic flow, different frameworks, mathematical definition of FL, and applications of FL, respectively. In section 2, this paper comprehensively discusses the key challenges of FL including a critical overview of recent approaches to address those challenges. Section 3 discusses some open research areas and finally, the conclusion is presented in section 4.

1. 1. Basic Flow of Federated Learning

Figure 1 illustrates the basic flow of FL. Here we typically assume that some clients want to collaborate for training a global model to perform some specific tasks. All participating devices collaborate with each other through a centralized server (aggregation server). In the first place, the centralized server forwards the copy of the global model to all active participants (active devices), then these devices train their copy of the global model 0n their

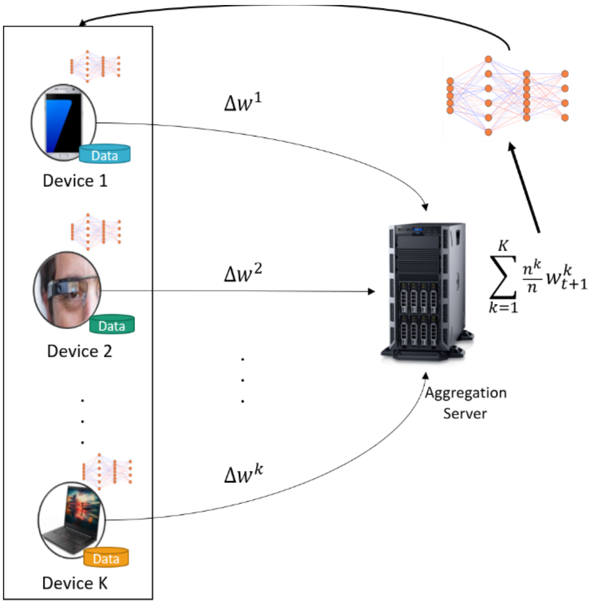


Figure 1. The basic architecture of Federated Learning

private data and send this updated model back to the server. After receiving all model updates from clients, a centralized server typically performs weighted aggregation on these local models' updates (parameters/gradients) to update the primary global model. Subsequently, the centralized server again sends this updated global model to all active clients, so active clients retrain this model. This process typically continues until the global model is converged.

1. 2. Different Frameworks To evaluate properly the performance of novel proposed solutions in federated learning, typically, a simulated environment is manually designed. For instance, datasets are manually split into different subsets in such a way these subsets mimic the behavior of Non-IID and unbalanced datasets. However, recently, different frameworks and benchmarks [35-44] have also been proposed for federated learning. Some of these frameworks also provide the federated datasets which fulfill the requirements of federated learning settings like distributed, unbalanced and Non-IID datasets. Similarly, some of these frameworks also provide the feature to compare different FL algorithms using different evaluation metrics. Though these datasets do not exactly mimic the real-world FL scenario still researchers can simulate their FL work using these datasets to mimic the behavior of near to real-world FL scenarios. A summary of these FL frameworks is given in Table 1.

1. 3. Definition of Federated Learning Suppose, K number of devices are participating in the federated learning process so dataset D is distributed among K devices as $D = \{D_1, D_2, D_3, .., D_k\}$ where each user i has a dataset $D_i (i \leq k)$, having n number of samples $\{(X_1^i, Y_1^i), (X_2^i, Y_2^i), (X_3^i, Y_3^i), \dots, (X_n^i, Y_n^i)\}$. Here each$X_i^k \in X_i$. Typically, the objective of federated learning is to minimize this objective function:

$$\min_{w} F(w) \quad where \quad F(w) = \sum_{k=1}^{K} \frac{n^k}{n} F_k(w) \tag{1}$$

TABLE 1. Some frameworks for federated learning Simulation

Approaches	Reference	Key Idea
TensorFlow Federated	[35]	An open-source framework that provides a platform for research experiments and large-scale simulation in FL. Additionally, it provides various federated datasets.
LEAF	[36]	Provide many datasets for benchmarking federated learning, MTL and Meta-Learning
PySyft	[37]	An open-source framework that combines FL and differential privacy, and integrates with deep learning frameworks like Keras, Tensorflow, or PyTorch to provide secure and private computations.
FATE	[38]	An open-source project which provides a secure computing framework
PaddleFL	[39]	Open-source framework based on PaddlePaddle which facilitates the researchers to compare different FL algorithms and deploy the FL system easily in large-scale distributed clusters.
NVIDIA Clara	[40]	A healthcare application framework but also provides full-stack GPU-accelerated libraries and SDKs to support FL.
OWKIN	[41]	Building a very large collaborative research network in health based on FL. They have also partially open-sourced their code for collaboration with other researchers and organizations.
Fedeval	[42]	It supports FedSGD and FedAvg algorithms. Use different evaluation features like accuracy, communication, time consumption, privacy, and robustness (ACTPR).
Fedml	[43]	Provide support for many machine-learning and FL algorithms. It supports mobile on-device training, distributed training, and standalone simulation.
OARF	[44]	Try to imitate real-world data distribution by collecting public datasets from distinct sources.

$$where \quad F_k\ (w) = \frac{1}{n_k}\ \sum_{i\in|n_k|} f_i(w)$$

Here n is the total number of samples in D while n_k is the total number of training samples in D_k on device $k \in K$. Here $f_i(w)$ represents the local objective function of each client k at a sample $i \in |n_k|$.

1. 4. Applications of Federated Learning Though the primary focus of federated learning is to learn from smart devices (cross-device) including smartphones, smart glasses, smartwatches, etc. where devices might have immensely valuable data but due to privacy concern. This enormous amount of valuable data is unavailable to train models for more accurate and efficient AI applications. But federated learning concept could be further extended to various organizations (cross-silo) like banks, hospitals which can collaborate with each other while preserving the confidentiality of users' data. For example, hospitals can securely collaborate with other hospitals or smart devices, containing health-related data, to train a shared global model for diagnosis or treatment of various medical disorders. Similarly, banks can also collaborate with each other, preserving the privacy of users' data, to train a shared global model for detecting the scam or fraudulent transactions.

There are many potential application areas [45-58] for federated learning including healthcare, sentiment analysis, recommendation systems, voice recognition, face detection, next-word prediction, predicting users' activities, autonomous vehicles, etc.

2. KEY CHALLENGES OF FEDERATED LEARNING

Federated learning has many unique challenges which typically make it different from traditional distribution optimization. These key challenges typically include model heterogeneity, statistical heterogeneity, communication (including system heterogeneity), and privacy. The following subsections concisely yet comprehensively explain these challenges with a critical review of recent approaches to handle these challenges.

2. 1. Statistical Heterogeneity In decentralized settings we typically assume, there is no centralized server to properly manage the distribution of data. Thus, it is very likely that clients would have highly unbalanced and Non-IID data as each device or user may have distinct preferences. For instance, let suppose there are two devices that want to collaborate in the federated learning scenario to train a unique global model (e.g., image classifier for fruit categories) while preserving the privacy of their data. Suppose the first device contains 50 samples of each of two classes (say apple and banana) while the second device contains 100 samples of each two classes (say orange and mango) so here these devices have unbalanced (varied number of samples for each class) and Non-IID (samples of distinct classes) data. Figure 2 presents an example of statistical heterogeneity.

FedAvg [17], a state of art algorithm based on SGD, shows that it can handle a certain amount of Non-IID data

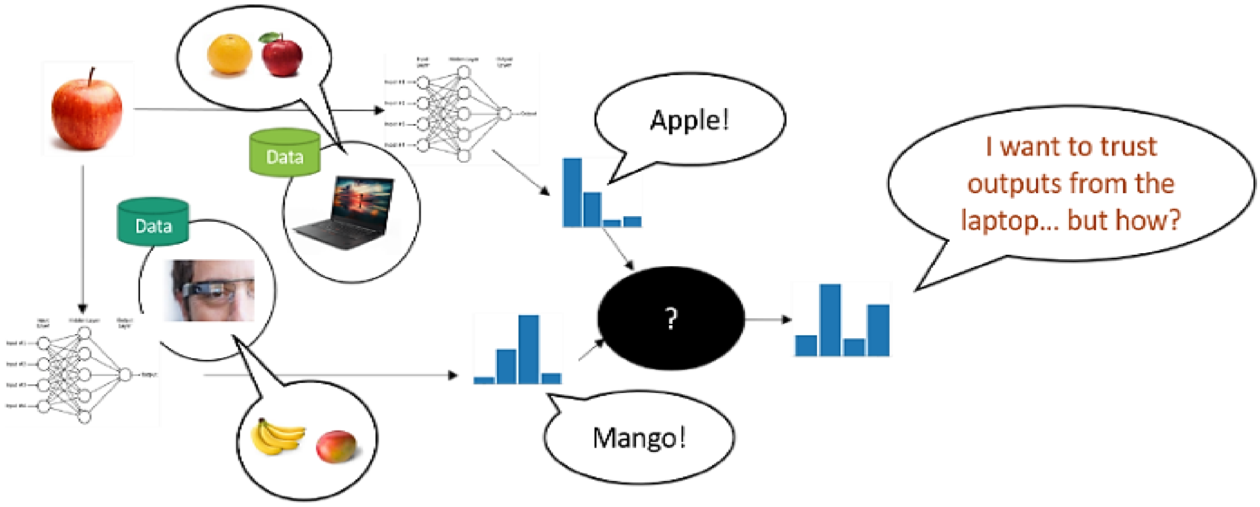


Figure 2. An example of Non-IID data. Suppose there are two devices, a laptop, and smart glasses, having their private data and a local model. The laptop has samples of orange and apple while smart glasses have samples of banana and mango. Here, they have Non-IID data distribution as they have samples of different classes. Now, suppose we are training a classifier to predict the fruit category. These local models would be well-trained to predict about classes for which they have training samples but not otherwise. For instance, if a sample of apple is provided to laptop-model then it would be reasonably confident that it is apple but if the same sample would be provided to the smart glasses model, then it would wrongly classify it like banana or mango. Now, the server needs to perform aggregation (as required in FL) but as you can see the server cannot take simple aggregation, rather it would like to give more preference to laptop output. But How? This is a simple example of a Non-IID problem in the context of FL

TABLE 2. Some approaches addressing statistical heterogeneity

Approaches	Reference	Key Idea
Sharing some data	[59]	Using the conditional GAN to produce the missing label samples on devices by leveraging the private data of devices.
	[60]	Reducing the weight divergence b/w different distributions by leveraging the globally shared data
	[61]	Devices remove their distribution imbalance by data augmentation before local training.
Multitask Learning	[62]	Proposed MOCHA and show that MTL is a natural way to handle the statistical challenge.
	[63]	Employ the concept of Bayesian network and perform variational inference during learning.

but Smith et al. [61] have shown empirically that for high skewed Non-IID data, the performance of the convolutional neural network, trained using FedAvg can drop reasonably by 51% on CIFAR-10, 11% on MNIST and 55% for keyword spotting datasets. Therefore, FedAvg does not effectively handle the skewed Non-IID data which is natural and expected data distribution in the federated learning setting.

Some researchers proposed the idea to share some of the local data of devices or share some proxy data to handle the statistical heterogeneity [59, 60] to make the data distributions of devices as IID. Like, Zhao et al. [60] show that accuracy reduction, in the case of Non-IID data, could be attributed to *weight divergence* when two different training processes having the same weight initialization get different weights. They propose that if we can leverage the globally shared data (having uniform distribution over all classes) by distributing it to all clients, then it can reduce the weight divergence between distribution on the different devices. This weight divergence could be quantified using EMD (Earth Moving Distance) and in return, it would increase the accuracy of the model. Their approach does not look much practical as arranging and communicating uniform distribution of data over all classes could be challenging and can create overhead for communication.

Jeong et al. [59] proposed FAug (Federated Augmentation) that uses the concept of conditional Generative Adversarial Network (GAN) to produce the missing label samples on client devices by data augmentation. Where each client is required to identify and upload the missing target labels, in its distribution, to the server. Server oversamples these target labels to train the conditional GAN.

Finally, all devices download this trained GAN to produce missing target labels in their distribution. As target labels of each device may reveal some private and sensitive information with the server or with other devices (which have GAN, trained on all devices' data and that could be used to infer the other's target labels), therefore it requires all devices to additionally upload the redundant samples, other than target labels, on the server to handle the privacy issue at the cost of extra communication overhead. However, this method works with the assumption that client devices would agree to share their private data with the server. This seems an almost impractical solution and violates the key idea of FL i.e., privacy.

Some researchers [62-65] have shown that the natural way to address the statistical challenge (Non-IID) of data is Multitask Learning (MTL) where the goal is to learn from each node, having separate but related models, simultaneously. Here, each node represents a task that possesses their private data and the goal is to learn from these related but different tasks. Like Smith et al. [62] used the MTL in the federated learning setting. In MTL, an additional term is included in the loss function to model the relationship among tasks. They used the correlation matrix to measure the client similarity and trained separate but related models for each device (task) using a shared representation on the server. However, their method only works for convex optimization problems and is not scalable to a large population. Furthermore, Lim et al. [66] argue that this approach is not much suitable in federated learning scenarios when a specific task (model) doesn't possess its local data or may have very few training samples.

Corinzia et al. [63] employ the concept of a Bayesian network to connect all clients with the server and perform variational inference during learning. Their method can properly handle the non-convex problem, but it is much costly to scale it to a vast federated network as it refines the client models sequentially.

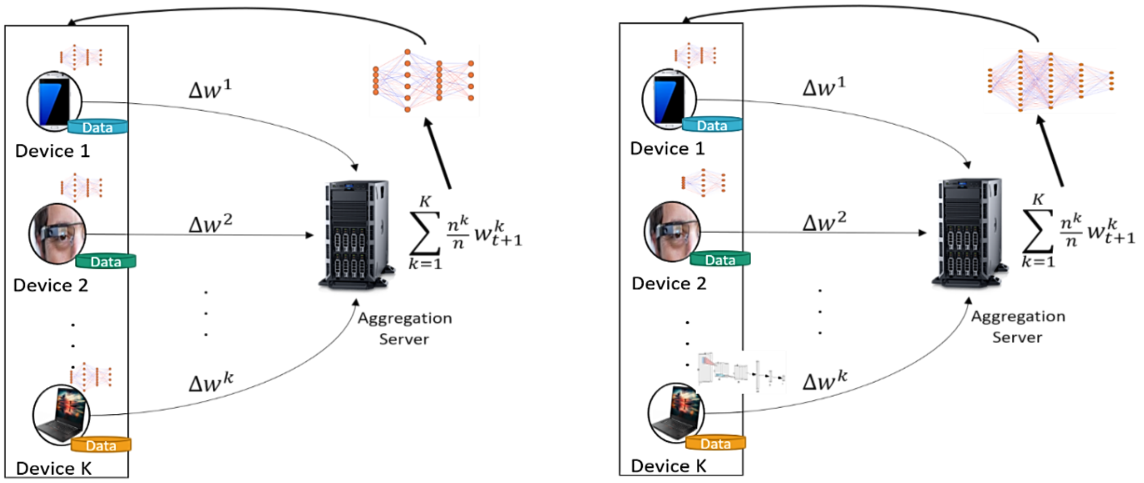
Duan et al. [61] revealed that model performance could also deteriorate due to global imbalance (when local distributions of data across all clients have class imbalance). It first removes the global imbalance by data augmentation where all devices first share their data distribution with the centralized server. Then, before performing local model training, each device first performs data augmentation on imbalance classes to make a balanced distribution. Subsequently, it employs the concept of mediators to combine training samples of relevant devices (selection is performed by calculating KL divergence between local and uniform distribution) based on their distributions to make it a uniform distribution. So, finally, this combined training (model) is shared with the global server for federated aggregation.

Recently, some researchers [67-71] have identified the limitations of the standard FedAvg algorithm, particularly, when clients have statistical heterogeneity. Lim et al. [66] questioned the performance of the standard FedAvg algorithm and suggested that standard aggregation is probably not the best aggregation way. By using the Mutual Information (MI) and different distance metrics, they demonstrate that with the increase in the number of iterations, correlation (MI) increases but in parallel, the distance of parameters also increases. Similarly, Xiao et al. [67] mention the three limitations of FedAvg i.e. 1) it cannot be applied on non-differentiable methods, 2) it usually requires many communications rounds, and 3) it is primarily designed for the cross-device setting. While addressing these limitations, they propose FedKT for cross-silo scenarios which can learn from both differentiable and non-differentiable models. Li et al. [68] explain that due to permutation invariance of NN, simple model parameter aggregation (FedAvg) may have a very negative impact so they propose PFNM, a probabilistic Federated Neural Matching algorithm that performs the matching among clients' NN neurons before averaging them. Yurochkin et al. [69] further extend this approach and propose a layer-wise matching approach (FedMA) and apply this approach to modern CNNs and LSTMs. Wang et al. [70] try to reduce Aggregation Error (AE) by constructing a definitely convex global posterior using a Gaussian product method to obtain the global expectation and co-variance by multiplying local posteriors. On the client-side, they proposed a new Federated Online Laplace Approximation (FOLA) method to obtain online local posterior probabilistic parameters which can directly be leveraged in the FL framework. Table 2 shows some recent approaches addressing statistical heterogeneity.

2. 2. Model Heterogeneity/Personalization As shown in Figure 3, typically, system heterogeneity is defined as where devices possess varied computational resources like different memory, processor, battery limit, active time, etc. More specifically, in model heterogeneity cases, due to varied computational resources and different business needs (trade-off between speed and accuracy), it is intuitive that devices may have the varying size of deep networks (different no. of layers) or may have completely different network architectures like some devices may be using CNN, some device may be using ResNet while some devices may opt for Inception.

Having the same model architecture would not only overburden the communication (already facing high communication challenges in federated learning) but would also increase the computation complexity for devices where low resourced devices may result in the form of stragglers or staled data. Probably, some devices might possess immensely valuable data but unable to train the same complex model. Therefore, it is intuitive that models should possess the proper number of nodes in their output layer to avoid unnecessary computation and communication overhead.

In this section, we focus on model heterogeneity, and regarding system (hardware) heterogeneity, the comprehensive discussion is performed in section 2.3. Table 3 presents some recent works to address model heterogeneity.



(a) Standard FL where all users have same model architecture

(b) More Practical scenario where different users might have heterogeneous model

Figure 3. This illustrates the difference between (a) standard federated learning where all devices are required to have the same model architecture and (b) a More practical scenario of federated learning where different users might have different model architectures based on their computational resources and business needs

TABLE 3. Some approaches addressing model personalization

Approaches	Reference	Key Idea
Retraining global model	[72]	Retrain global model on client's private data.
Transfer Learning	[73-75]	Model personalization using transfer learning.
Meta Learning	[76-79]	Typically, the model is made adaptive by training it on multiple tasks.
Multitask Learning	[63]	Employs Bayesian Network and performs variational inference during learning.
	[62]	Extends MTL and uses a modified loss to represent relationships.
Knowledge Distillation	[80]	Allows the model heterogeneity.
	[81]	Address the model heterogeneity with Non-iid data.

We can divide the model heterogeneity into two general categories 1) where different models need to be personalized based on different geographical or personal preferences like in the case of next word predication, for a sentence "I love to visit ….", there would be customized predictions for different users living in separate geographical location or with distinct preferences. 2) where various models might have diverse architecture due to varied computational resources or different business needs. For instance, one might be using CNN with 5 layers, the second is using CNN with 10 layers, the other is using Random Forest, etc.

The primary goal of federated learning is to train a unique global model by leveraging the multiple clients' private data and their computational resources. In FL, it is assumed that all devices would be able to train the same copy of local solver (same model architecture) on their private data, and then these model's updates are sent to the aggregating server which further aggregates these updates.

Moreover, standard FL works with the assumption that all clients would possess reasonable computational and communicational resources to train the same model architecture. However, in the real scenario, this key assumption of FL does not seem logical as more complex deep learning models are being developed to achieve more accurate performance on real-world tasks. Furthermore, intuitively, all devices cannot be capable to train the same complex model. This incapability of models could be due to limited computational resources of devices or due to varying business needs (trade-off between accuracy and speed) or to reduce the computational and communication overhead.

Due, to such primary assumption of FL, most of the work has been performed with the same assumption of homogeneous models and most of the work has been performed for category 1 of model heterogeneity (model personalization based on personal preferences or distinct geographical locations) and very few works have addressed this potential problem of category 2 of model heterogeneity (clients contain diverse model architecture) and in a significantly limited fashion. To make a global model personalized, most personalization techniques suggest retraining the (collaboratively trained) global model on the users' local private data [72]. Some researchers have proposed model personalization approaches using transfer learning [73-75]. In transfer learning, usually, the last layers of a trained model are replaced with new layers to leverage the learned knowledge of the trained model on some new tasks. Some researchers suggest freezing the initial layers of a trained global model and retrained only the last few layers on local private data of individual clients.

Recently, some researchers [76-79] have also leveraged meta-learning to solve the personalization problem. Meta-learning is generally defined as "Learning to Learn" where a model is made adaptive by training it on multiple tasks in such a way that it can learn new tasks by providing very few examples of new tasks.

Multitask learning [82] has also been widely used by different researchers [62-65, 83, 84] to address the model personalization challenge. They leverage the distributed Multitask Learning to train separate but related models. They tried to train a personalized model for each distribution (as in FL, we assume the Non-IID distribution). Like, Smith et al. [62] extended the MTL in the federated learning setting. To model the relationship among tasks, they include an additional term in the loss function. They use the correlation matrix to measure the client similarity and train separate but related models for each device (task) using a shared representation on the server. However, their method only works for convex optimization problems and is not scalable to a massive population. Similarly, Corinzia et al. [63] employ the concept of a Bayesian network to connect all clients with the server and performs variational inference during learning. Their method can handle the non-convex problem, but it is much costly to scale it to a large federated network as it refines the client models sequentially. In addition to limitations of scalability and feasibility, these approaches do not address the other model heterogeneity scenarios like models having entirely diverse architectures and having different output layers.

Knowledge distillation [85] is a technique to distill the knowledge from a pre-trained cumbersome (teacher) model into a small (student) model so that the student model can also mimic the behavior of the teacher model. Li et al. [80] leverage the knowledge distillation to allow clients to use customized local models having a different number of layers. Therefore, it allows the clients to use diverse model architecture while collaborating in Federating Learning. But they have not evaluated their method with totally different model architectures.

Recently, Ma et al. [81] have proposed a very effective adaptive distillation approach where they address the model heterogeneity problem using Non-iid data. More specifically, they trained the local model using their own private labeled data whilst trained the global model using unlabeled Non-iid public data.

2. 3. Communication One of the primary challenges of federated learning is the Communication overhead of downloading the global model's parameters (or gradients) from the centralized server and then uploading the trained model's parameters (or gradients) back to the server in each round of communication. This communication overhead is proportional to the model size means for large-scale models; the number of model parameters could be in millions. More specifically, it becomes more challenging and extremely costly for devices having limited bandwidth and intermittent network connections. Because, intuitively, smart devices might possess good computational capabilities. However, these devices are likely to have different network bandwidth like some devices may support significantly efficient networks say 4G, 5G, or Wi-Fi, and some devices may only support significantly poor network connections. Similarly, some devices may bear good and stable connections whilst mostly mobile devices may bear unstable and intermittent limited connections in the real scenario. These network limitations including unpredictable network interruption could also result in the form of stragglers and passive sampling.

Researchers of naïve federated learning approach [16] have shown that the size of the update (parameters) would be independent of local training data of individual devices and the global model can be trained in few communication rounds. Therefore, as compared to sharing training datasets of all active devices with the centralized server, FL reduces the communication cost by order of magnitude. However, due to unpredictable network limitations of participating devices in FL, it is still an open challenge to reduce the communication cost during each round of communication with the centralized server. They have also proposed different methods like sketched updates to reduce the communication cost in the order of magnitude as compared to sharing data.

Usually, in centralized learning approach, there is less communication cost as compared to computation cost but, on the contrary, in decentralized learning approach, communication cost becomes more challenge as compared to computation cost because modern smart devices, generally, have more high processing power while having very less training data (as a fraction of total dataset) to be trained [86]. Therefore, we might, loosely, say that computation almost becomes free as compared to communication for many model types.

Intuitively, there are two straightforward approaches to reduce this communication cost; one is to add more parallelism by including more clients in each round of communication. The second approach is to add more complex computation on each device like perform many gradient steps on each client, instead of one, in each communication round. McMahan et al. [17] have empirically shown that adding more clients results in diminishing return after a particular limit but, in contrast, adding more complex computation produces more promising results.

Many works have been performed to reduce this communication cost of transferring the large weight matrices of deep networks, to handle unexpected interruption or dropout of participating devices, and synchronization latency caused by the computing power and network connectivity constraints. Asynchronous SGD [18, 87-93] also tries to handle this communication bottleneck. It accelerates the training process by updating the parameters immediately after a computing node has sent its gradients (asynchronous communication) instead of waiting for all computing nodes to send their gradients before updating parameters. Albeit it speeds up the training process and handles the stragglers' problem, but it comes with the staled gradient problem which could also affect the accuracy of the model. Some work has been performed to address the issues of active sampling where in each round of communication, clients fulfilling a particular criterion is selected rather than passive sampling where there are no criteria to select the effective clients like Nishio et al. [103] proposed a resource-based active device sampling technique for heterogeneous clients where the FL server first sends the resource request to maximum clients to get information about their available resources.

Subsequently, only those clients are selected who can complete the training process within a specified threshold. By selecting the maximum number of clients in each round, it assists the global model to attain high accuracy. But they do not address the data distribution issue like some devices, having high computational resources, might not have much data.

To reduce communication bandwidth, model compression schemes [95, 96] are also being used to reduce the model size, for communication, using different techniques. But most of the compression schemes work in the data center environment and system challenges of federated learning like participation of devices could be low, Non-IID data and local update schemes introduce new challenges for these techniques. Some researchers have proposed gradient quantization methods [16, 98, 99], where gradients are quantized to low precision values to force the updating model to be sparse and low rank. Some of these approaches [98] could be difficult to extend to federated learning as errors accumulated locally may become staled if devices could not participate frequently in communication rounds. Some researchers have also proposed the gradient sparsification methods [100] where all gradients are not sent to the server rather only those gradients larger than a predefined specific threshold are sent to the server. As it could be much challenging to choose a correct threshold so some other techniques in gradient sparsification are also proposed like gradient dropping [101] to drop gradients after a particular absolute value or automatically tuning the compression rate based on the activity of local gradients [104]. Lin et al. [102] proposed a Deep Gradient Compression; a gradient compressing technique that employs various methods like momentum correction, local gradient clipping, momentum factor masking, and warm-up training to reduce the communication bandwidth (which could also result in the staleness problem) by two-order of magnitude without compromising the accuracy of the model. The author discussed the scenario of federated learning in his paper, so this technique could be extended in the federated learning setting.

Some researchers have leveraged distributed multi-task learning [62, 64, 83, 94, 105, 106] for communication efficient learning in the distributed environment but most of them do not mitigate the system challenges of the federated learning environment. Like Baytas et al. [94] allows for asynchronous updates to handle stragglers but doesn't address the fault-tolerance problem and it gives the convergence guarantee on the assumption of bounded delay which is not possible in federated learning setting where devices may experience significant delays or drop out completely due to sudden network interruption. Liu et al. [83], extend the distributed framework COCOA [107] to learn the relationship among distributed tasks along with the predictive models for each task but they do not explore the federated settings and make the assumption that data distribution is balanced means each device would perform a similar amount of work.

Smith et al. [62] have proposed an alternative of FedAvg (standard federated learning algorithm), called MOCHA for federated multitask learning setting. They more significantly extend the communication efficient algorithm COCOA to address the problems of fault tolerance and stragglers in the federated learning setting. But the effectiveness of their problem is limited to only convex problems.

Jeong et al. [59], have employed the concept of online distillation in federated learning. Online distillation is a simple model compression technique, called Codistillation [108] to reduce the communication overhead by many orders of magnitude. In this approach, communication overhead does not depend upon the model size or dataset size rather it depends upon the output dimension of the model. It shows that they have reduced the communication overhead around 26x as compared to the standard Federated Averaging [17] approach where parameters or gradients of models are shared with the parameter server. Similarly, Guha et al. [97] proposed one-shot federated learning. They additionally use the concept of ensemble learning and knowledge distillation to reduce the communication for the convex optimization problem. Table 4 presents some recent works to address communication challenge.

TABLE 4. Some approaches addressing communication challenges

Challenge	Approaches	Reference	Key Idea	Target
Fault tolerance, Stragglers	Asynchronous SGD	[18]	Perform the aggregation immediately after receiving data from any device	Synchronization latency due to low resourced devices, stragglers
	MTL approaches	[94]	Asynchronous distributed MTL	Stragglers
		[62]	Proposed MOCHA, generalizing the COCOA	Communication cost, stragglers, fault tolerance
		[64]	Extend the communication efficient algorithm COCOA.	Fault tolerance, stragglers
Communication overhead	Model compression schemes	[95, 96]	Model compression schemes	Reduce communication overhead
		[59]	Co-distillation, a model compression scheme	Reduce communication overhead
		[97]	Proposed one-shot federated learning using ensemble learning.	Reduce communication overhead
	Gradient compression schemes	[16, 98, 99]	Gradients are quantized to low precision values	Reduce communication overhead
		[100]	Gradient sparsification; only gradients larger than a particular threshold are sent to the server.	Reduce communication overhead
		[101]	Gradient dropping; drop gradients after specific threshold value.	Reduce communication overhead
		[102]	Employ various gradient compression schemes like momentum correction, local gradient clipping, momentum factor masking, and warm-up training.	Reduce communication overhead

2. 4. Privacy/Security It is the primary motivation behind federated learning, where we want to learn from user's data while securing the privacy of users' data. Typically, we can categorize privacy issues in three subproblems [15] as *privacy of input data*, the *privacy of the trained model*, and *privacy of the model's output*. Federated learning naturally ensures the privacy of input data as it only works on learned parameters from the private data rather than the original raw data. Though FL, try to secure the private data of clients by sharing only the trained parameters; however, recent studies [109, 110] have revealed that valuable information about client's training data could be reasonably inferred from its trained model update (learned parameters) with very high accuracy of up to 90%. It clearly explains that despite not utilizing private data in FL, still, private information of clients is vulnerable. Therefore, many solutions have been proposed to address various kinds of vulnerabilities.

Typically, we can categorize security issues into three general categories including data poisoning, model poisoning, and evasion attacks. In a *data poisoning attack*, some adversary clients may intentionally use the malicious data samples to mislead the global model by providing their local models trained on malicious data. In a *model poisoning attack*, adversaries' target is to misguide the machine learning models to produce malicious results [111]. Model poisoning attacks can further be divided into two categories. targeted and un-targeted adversarial attacks. *Targeted attacks*, also called backdoor attacks, usually do not compromise the overall accuracy of the global model rather just focus on some specific classes or examples. On the other hand, *un-targeted attacks* try to mislead the whole global model [112, 113]. In *evasion attacks*, also called inference-time attacks, adversaries puzzle the deployed machine learning model by providing such misleading and modified samples which seemingly look like original test samples [114].

To preserve the information from other clients, one of the most commonly applied privacy-preserving techniques is Differential Privacy (DP) [15, 115-117] due to its simple algorithm and relatively low communication overhead. In this technique, some noise is added in trained parameters of the model before uploading them to the aggregation server to make it impossible for the third party to distinguish the individuals. Abadi et al. [115] proposed a technique for deep learning algorithms where it includes a noise to trained parameters, before forwarding them to the server, using a Gaussian distribution. Similarly, Geyer et al. [116] further enhanced this method by introducing two steps; in the first step, the server selects some random clients to participate in a communication round, and then only those randomly selected clients would include noise in their trained parameters using Gaussian distribution before forwarding them to server. Hence, in this way, other participating clients would be unable to know which clients are participating in this round. so would not be able to infer the information from shared parameters of the global model. Differential privacy comes with the cost of a reduction in the accuracy. There exists a tradeoff between model accuracy and differential privacy because when we include more noise to ensure more privacy, it results in the reduction of model accuracy significantly. In the same way, we also need to consider the tradeoff between device performance and DP as system computing resources would be required for applying DP.

Another effective approach for privacy-preserving of distributed datasets is Secure Multiparty Computation (SMC) [118-123] where multiple parties collaboratively compute a function using their inputs without revealing their private inputs to other parties. This additionally requires extra computation and communication overhead. In addition, in this approach, typically, a minimum portion of users' data must be shared.

Similarly, fully homomorphic encryption and its variants [52, 123-127] are also being employed to improve the security of trained models. In this method, participating clients can only see the encrypted data and they need to perform some computation on this encrypted data. Results are sent to the owner and usually, only the owner has a private key to decrypt the data. Usually, homomorphic encryption can be divided into three categorized based on the number of operations allowed to perform on encrypted data. 1) Partially homomorphic encryption (PHE) 2) Somewhat homomorphic encryption (SWHE) and 3) Fully homomorphic encryption (FHE). Acar et al. [128], the authors have presented a comprehensive survey on homomorphic encryption schemes.

Bonawitz et al. [122] proposed a secure, failure-robust, and communication efficient protocol. The goal is to learn from a significant number of mobile devices by aggregating their contribution in a secured manner to prevent identifying the individual's contribution in collaborative learning. Though they proposed their work as a general secure communication protocol, however, they suggest that their method could be employed in federated learning settings where clients share their model rather than their private data, and these models need to be aggregated securely.

Another potential approach to address the privacy issue of FL is proposed by Mandal et al. [123] where authors guarantee the model and data privacy for user and server using homomorphic encryption. In addition, Xu et al. [129] proposed VerifyNet, which primarily addresses two problems 1) How to protect user's privacy during the training process and 2) How to trust the results from the server i.e. verification of aggregated results from the server. In their proposed approach, the server is supposed to provide the proof of correctness to all clients and there is almost no possibility of forging proof as the adversary needs to solve the adopted NP-hard problem to create a forging proof.

Chen et al. [52] proposed FedOpt to address the privacy and communication efficiency challenges of FL. They also design a Sparse Compression Algorithm (SCA) for communication efficiency and then integrate the additively homomorphic encryption with differential privacy to prevent data from being leaked. An overview of some recent approaches is given in Table 5.

3. FUTURE DIRECTIONS/OPEN CHALLENGES

In recent years, federated learning has emerged as an extremely promising domain to collaboratively learn from highly unbalanced, massively distributed Non-IID data. A lot of research work has been performed to address the numerous challenges of federated learning. However as critically discussed in section 2, there are many open research problems to be considered for more practical scenarios of federated learning. Some of them are discussed below.

3. 1. Supervised/Unsupervised Training Most of the work done in the field of FL is based on supervised learning where it is assumed that all devices have correct labels against their training samples. However, in a real scenario, some devices might contain unlabeled data or might bear incorrect labels against their data. Similarly, some devices might consist of missing classes in their training data. Therefore, it would be challenging for the global model to learn effectively or indicate similar confidence to all participating clients.

Recently, some researchers [130, 131] have leveraged semi-supervised and unsupervised learning in FL. In semi-supervised learning, it is supposed that very few label data are available, so a model is trained on both (available) labeled data and (public) unlabeled data. While in unsupervised learning, it is supposed that no label data is available to train the model, so the model is trained solely on unlabeled data. Recently, contrastive learning [132-135] has become a hot research direction to learn the representations of unlabeled data by training the model on the unlabeled data. Contrastive learning (also referred to as self-learning) is a pre-training process where the model tries to learn similar and distinct data samples in an unlabeled data distribution. Recently, Chen et al. [134] present an effective contrastive learning framework (simCLR) that outperforms the other state-of-the-art algorithms [132, 136, 137]. Very few works have been carried out to apply the FL with semi-supervised and unsupervised data. Therefore, it is still an open research area to be explored in FL settings.

3. 2. Synchronous/Asynchronous FL Typically, the most standard approach in current FL work is synchronous FL where the aggregation server waits for local updates from all participating devices. Then server performs aggregation on all these updates. However, it can result in a straggler effect where some devices might be significantly slower due to their local capacities like limited computational resources, limited power, or low bandwidth internet. Another potential direction is to use FL in the asynchronous mode [87-93] where the aggregation server would perform aggregation as it receives updates from participating devices. Though it could handle the straggler effects and could provide the flexibility to devices to join the FL process in halfway; however, it could result in the form of staled updates; where some devices with low capabilities might send the outdated outputs to the server which can affect the convergence of the global model. This asynchronous scheme has been applied successfully in centralized distributed computing where the bounded delay is expected. However, in the practical scenario of FL, it could be almost impossible to set a fixed bounded delay because, in FL settings, the delay could be up to a couple of hours or days.

Furthermore, as discussed above, that practical FL also experiences the challenges of system heterogeneity due to varied computational resources, memory, network infrastructure. Therefore, there are strong possibilities that some devices might drop during the FL process.

TABLE 5. Some approaches addressing privacy/security challenge

Approaches	Reference	Key Idea
Differential privacy	[115]	Include noise to trained parameters before aggregation.
	[116]	Two-fold protection, noise is added to random clients before aggregation.
	[52]	Integrate the additively homomorphic encryption with differential privacy.
Secure Multiparty Computation	[118-121]	Multiple parties compute collaboratively compute a function using their inputs.
	[122]	Proposed MOCHA and show that MTL is a natural way to handle the statistical challenge.
	[123]	Train their models based on an additive homomorphic encryption (HE) scheme and an aggregation protocol.
Homomorphic encryption	[52, 123-127]	Clients are required to perform some computations on encrypted data and only the owner can decrypt the data using the private key.

Therefore, FL systems should be capable to consider fault tolerance. Some researchers suggest permanently dropping such devices from the training process. However, it may result in the form of biased model training due to the loss of data samples possessing some specific characteristics. This drop-out (stateless) problem also led to other challenges of privacy and security where proposed solutions for privacy/security require stateful clients for validation and verification of trusted clients. Like, it could be more challenging for secure aggregation to work effectively if a significant number of clients suddenly leave the training process. Recently, some works have tried to address this issue like Wu et al. [89] proposed a semi-asynchronous learning client selection algorithm and a lag-tolerant mechanism to overcome different challenges like stragglers and model staleness in FL. Li et al. [93] introduce asynchronous updater to actively receive the unsynchronized local weights from stragglers. However, new algorithms and methods are required to address these challenges more effectively.

3. 3. Privacy/Security Privacy is the fundamental issue in FL and a lot of work has been done to address this issue but most of those approaches are not effectively applicable in real-world scenarios due to the limitation of computational and network resources including the availability of all devices in each round of communication (as discussed in section 2.3). For instance, secure aggregation is an effective approach to ensure privacy. Nevertheless, it has become a hurdle for other defense approaches to implement because the server can merely examine the aggregated outputs and usually unable to differentiate the outputs of clients.

As mentioned in section 2.4, there are many security challenges for FL including data poisoning, evasion attacks, and model poisoning [111-114]. Similarly, there are many other methods to target the FL system like some devices might work as free-rider [138]; where devices do not have relevant training samples but indeed want to get benefit from other devices. In such a way, it would equally compel other devices to provide more computational resources for the FL process. Contrary to centralized learning, it is challenging to figure out the trust mechanism for participating clients to identify whether clients are trustworthy or not. Though some approaches [139] have been proposed to address some of these issues, however, it usually requires a trade-off between performance and security of the FL system so still, there is a need for more robust and effective methods to address these challenges.

3. 4. Model Heterogeneity This paper also discusses some relatively new challenges for practical FL as discussed in section 2.2. Though model personalization is not a new research area, and some research works [62-65, 73-79, 83, 84] have been done in this area but very few works [80, 81] have been done to address the model heterogeneity challenge where devices might have entirely different model architectures based on their computational resources or business needs. This scenario becomes more challenging if we combine the missing classes problem with model heterogeneity.

3. 5. Different Federated Aggregation Algorithms As discussed in section 2.1, recently, some researchers have identified a relatively new challenge in FL where they demonstrate that the standard FL aggregation algorithm (FedAvg) is not the most effective method for clients' model aggregation particularly when clients have different data distributions. In the same way, FedAvg has many limitations including its applicability limitation to non-differentiable methods and its computation and communication overhead. Although many works have been, recently proposed to address the limitations of FedAvg including many alternatives of FedAvg like FedProx [28], FedMA [70], Scaffold [140]. Nevertheless, they still experience some limitations and can only be applied with some strong assumptions. Therefore, there is a need for some more robust approaches to address such challenges.

4. CONCLUSION

Federated learning has emerged as the de facto decentralized learning framework in privacy-preserving scenarios where the training data is not directly accessible. In federated learning, some devices train a statistical model on their private data and share only this trained model with centralized/aggregation server for collaborative learning. Though, federated learning is proven to be immensely beneficial in various domains including cross-device and cross-silo. But it equally has many challenges including system heterogeneity, statistical heterogeneity, communication, security, privacy, and model heterogeneity/personalization. This article briefly provides an overview of federated learning including its potential application areas. Subsequently, it comprehensively discusses its unique key challenges and a critical review of recent approaches to address these key challenges. Some relatively new challenges, like model heterogeneity and (global model) aggregation error, are also discussed in detail with potential approaches to address these issues. Furthermore, deriving from the key challenges' discussion, some open research areas are also discussed, in section 3, to be explored by the federated learning research community.

5. ACKNOWLEDGMENT

This research is sponsored by research grant (Malaysia Research University Network) 203.PKOMP.6777003.

6. REFERENCES

1. Taigman, Y., Yang, M., Ranzato, M.A. and Wolf, L., "Deepface: Closing the gap to human-level performance in face verification", in Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition, IEEE, (2014), 1701-1708.
2. Simard, P.Y., Steinkraus, D. and Platt, J.C., "Best practices for convolutional neural networks applied to visual document analysis", in Proceedings of the International Conference on Document Analysis and Recognition, ICDAR, (2003), 958-963.
3. Salehi, S.M.M. and Pouyan, A.A., "Detecting overlapping communities in social networks using deep learning", ***International Journal of Engineering Transactions C: Aspects***, Vol. 33, No. 3, (2020), 366-376, DOI: 10.5829/ije.2020.33.03c.01.
4. Hannun, A., Case, C., Casper, J., Catanzaro, B., Diamos, G., Elsen, E., Prenger, R., Satheesh, S., Sengupta, S., Coates, A. and Ng, A.Y., *Deep speech: Scaling up end-to-end speech recognition*, in *ArXiv*. 2014.
5. Gheitasi, A., Farsi, H. and Mohamadzadeh, S., "Estimation of hand skeletal postures by using deep convolutional neural networks", ***International Journal of Engineering, Transactions A: Basics***, Vol. 33, No. 4, (2020), 552-559, DOI: 10.5829/ije.2020.33.04a.06.
6. Shaeiri, Z. and Kazemitabar, S.J., "Fast unsupervised automobile insurance fraud detection based on spectral ranking of anomalies", ***International Journal of Engineering, Transactions A: Basics***, Vol. 33, No. 7, (2020), 1240-1248, DOI: 10.5829/ije.2020.33.07a.10.
7. Savadi Hosseini, M. and Ghaderi, F., "A hybrid deep learning architecture using 3d cnns and grus for human action recognition", ***International Journal of Engineering, Transactions B: Applications***, Vol. 33, No. 6, (2020), 959-965, DOI: 10.5829/ije.2020.33.05b.29.
8. Sezavar, A., Farsi, H. and Mohamadzadeh, S., "A modified grasshopper optimization algorithm combined with convolutional neural network for content based image retrieval", ***International Journal of Engineering, Transactions A: Basics***, Vol. 32, No. 7, (2019), 924-930, DOI: 10.5829/ije.2019.32.07a.04.
9. Ghanbari Sorkhi, A., Hassanpour, H. and fateh, m., "Improvement of the r-fcn's deep network in object detection and annotation", ***Journal of Machine Vision and Image Processing***, Vol. 6, No. 2, (2020), 43-59.
10. Poushter, J. "Smartphone ownership and internet usage continues to climb in emerging economies." Pew research center 22, no. 1 (2016): 1-44.
11. Neuromation, *What's the deal with "ai chips" in the latest smartphones?* 2018.
12. Blackmer, W.S., *Eu general data protection regulation*. 2018.
13. KPMG, *Overview of china's cybersecurity law. Kpmg advisory (china) limited*. 2017.
14. *California privacy rights act | californians for consumer privacy*. 2020.
15. Shokri, R. and Shmatikov, V., "Privacy-preserving deep learning", in Proceedings of the ACM Conference on Computer and Communications Security, New York, USA, ACM Press, (2015), 1310-1321.
16. Konečný, J., McMahan, H.B., Yu, F.X., Richtárik, P., Suresh, A.T. and Bacon, D., *Federated learning: Strategies for improving communication efficiency*, in *ArXiv*. 2016.
17. Brendan McMahan, H., Moore, E., Ramage, D., Hampson, S. and Agüera y Arcas, B., "Communication-efficient learning of deep networks from decentralized data", in Proceedings of the 20th International Conference on Artificial Intelligence and Statistics, AISTATS 2017, (2017).
18. Dean, J., Corrado, G.S., Monga, R. and Chen, K., "Large scale distributed deep networks", in Neural Information Processing Systems'12, (2012), 1223-1231.
19. Shamir, O., Srebro, N. and Zhang, T., "Communication-efficient distributed optimization using an approximate newton-type method", in 31st International Conference on Machine Learning, ICML 2014, (2014), 2665-2681.
20. Reddi, S.J., Konečný, J., Richtárik, P., Póczós, B. and Smola, A., *Aide: Fast and communication efficient distributed optimization*, in *ArXiv*. 2016.
21. Ma, C., Konečný, J., Jaggi, M., Smith, V., Jordan, M.I., Richtárik, P. and Takáč, M., "Distributed optimization with arbitrary local solvers", ***Optimization Methods and Software***, Vol. 32, No. 4, (2017), 813-848, DOI: 10.1080/10556788.2016.1278445.
22. Yang, Q., Liu, Y., Chen, T. and Tong, Y., "Federated machine learning: Concept and applications", ***ACM Transactions on Intelligent Systems and Technology***, Vol. 10, No. 2, (2019), 1-19, DOI: 10.1145/3298981.
23. Kulkarni, V., Kulkarni, M. and Pant, A., "Survey of personalization techniques for federated learning", in Proceedings of the World Conference on Smart Trends in Systems, Security and Sustainability, WS4 2020, Institute of Electrical and Electronics Engineers Inc., (2020), 794-797.
24. Xia, Y., "Watermarking federated deep neural network models", (2020),
25. Lyu, L., Yu, H., Zhao, J. and Yang, Q., Threats to federated learning, in Lecture notes in computer science (including subseries lecture notes in artificial intelligence and lecture notes in bioinformatics). 2020.3-16.
26. Li, Q., Wen, Z., Wu, Z., Hu, S., Wang, N., Li, Y., Liu, X. and He, B., *A survey on federated learning systems: Vision, hype and reality for data privacy and protection*, in *ArXiv*. 2019.
27. Cui, L., Yang, S., Chen, F., Ming, Z., Lu, N. and Qin, J., "A survey on application of machine learning for internet of things", ***International Journal of Machine Learning and Cybernetics***, Vol. 9, No. 8, (2018), 1399-1417, DOI: 10.1007/s13042-018-0834-5.
28. Li, T., Sahu, A.K., Zaheer, M., Sanjabi, M., Talwalkar, A. and Smith, V., *Federated optimization in heterogeneous networks*, in *ArXiv*. 2018.1-28.
29. Niknam, S., Dhillon, H.S. and Reed, J.H., "Federated learning for wireless communications: Motivation, opportunities, and challenges", ***IEEE Communications Magazine***, Vol. 58, No. 6, (2020), 46-51, DOI: 10.1109/mcom.001.1900461.
30. Tran, N.H., Bao, W., Zomaya, A., Nguyen, M.N.H. and Hong, C.S., "Federated learning over wireless networks: Optimization model design and analysis", in IEEE INFOCOM 2019 - IEEE Conference on Computer Communications, Institute of Electrical and Electronics Engineers Inc., (2019), 1387-1395.
31. Kairouz, P., Brendan McMahan, H., Avent, B., Bellet, A., Bennis, M., Bhagoji, A.N., Bonawitz, K., Charles, Z., Cormode, G., Cummings, R., D'Oliveira, R.G.L., Rouayheb, S.E., Evans, D., Gardner, J., Garrett, Z., Gascón, A., Ghazi, B., Gibbons, P.B., Gruteser, M., Harchaoui, Z., He, C., He, L., Huo, Z., Hutchinson, B., Hsu, J., Jaggi, M., Javidi, T., Joshi, G., Khodak, M., Konečný, J., Korolova, A., Koushanfar, F., Koyejo, S., Lepoint, T., Liu, Y., Mittal, P., Mohri, M., Nock, R., Özgür, A., Pagh, R., Raykova, M., Qi, H., Ramage, D., Raskar, R., Song, D., Song, W., Stich, S.U., Sun, Z., Suresh, A.T., Tramèr, F., Vepakomma, P., Wang, J., Xiong, L., Xu, Z., Yang, Q., Yu, F.X., Yu, H. and Zhao, S., *Advances and open problems in federated learning*, in *ArXiv*. 2019.

32. Aledhari, M., Razzak, R., Parizi, R.M. and Saeed, F., "Federated learning: A survey on enabling technologies, protocols, and applications", ***IEEE Access***, Vol. 8, No., (2020), 140699-140725, DOI: 10.1109/access.2020.3013541.

33. Li, T., Sahu, A.K., Talwalkar, A. and Smith, V., "Federated learning: Challenges, methods, and future directions", ***IEEE Signal Processing Magazine***, Vol. 37, No. 3, (2020), 50-60, DOI: 10.1109/msp.2020.2975749.

34. Mothukuri, V., Parizi, R.M., Pouriyeh, S., Huang, Y., Dehghantanha, A. and Srivastava, G., "A survey on security and privacy of federated learning", ***Future Generation Computer Systems***, Vol. 115, No., (2021), 619-640, DOI: 10.1016/j.future.2020.10.007.

35. TensorFlow, *Tensorflow/federated: A framework for implementing federated learning*, in *GitHub*. 2020.

36. Caldas, S., Duddu, S.M.K., Wu, P., Li, T., Konečný, J., McMahan, H.B., Smith, V. and Talwalkar, A., *Leaf: A benchmark for federated settings*, in *ArXiv*. 2018.

37. Ryffel, T., Trask, A., Dahl, M., Wagner, B., Mancuso, J., Rueckert, D. and Passerat-Palmbach, J., *A generic framework for privacy preserving deep learning*, in *ArXiv*. 2018.

38. FedAI.org, *Fate-federated ai ecosystem*. 2019.

39. PaddleFL, *Paddlefl*. 2019.

40. Clara, N., *Nvidia clara | nvidia developer*. 2019.

41. OWKIN, *Federated learning - owkin*.

42. Chai, D., Chen, K., Wang, L. and Yang, Q., *Fedeval: A benchmark system with a comprehensive evaluation model for federated learning*, in *ArXiv*. 2020, arXiv.

43. He, C., Li, S., So, J., Zhang, M., Wang, H., Wang, X., Vepakomma, P., Singh, A., Qiu, H., Shen, L., Zhao, P., Kang, Y., Liu, Y., Raskar, R., Yang, Q., Annavaram, M. and Avestimehr, S., *Fedml: A research library and benchmark for federated machine learning*, in *ArXiv*. 2020.

44. Hu, S., Li, Y., Liu, X., Li, Q., Wu, Z. and He, B., *The oarf benchmark suite: Characterization and implications for federated learning systems*, in *ArXiv*. 2020.

45. Li, S., Cheng, Y., Liu, Y., Wang, W. and Chen, T., *Abnormal client behavior detection in federated learning*, in *ArXiv*. 2019.

46. Liu, Y.Y.Y., Huang, A., Luo, Y., Huang, H., Liu, Y.Y.Y., Chen, Y., Feng, L., Chen, T., Yu, H. and Yang, Q., *Fedvision: An online visual object detection platform powered by federated learning*, in *ArXiv*. 2020.

47. Huang, L., Shea, A.L., Qian, H., Masurkar, A., Deng, H. and Liu, D., "Patient clustering improves efficiency of federated machine learning to predict mortality and hospital stay time using distributed electronic medical records", ***Journal of biomedical informatics***, Vol. 99, (2019), 103291, DOI: 10.1016/j.jbi.2019.103291.

48. Ramaswamy, S., Mathews, R., Rao, K. and Beaufays, F., *Federated learning for emoji prediction in a mobile keyboard*, in *ArXiv*. 2019.

49. Jiang, J., Ji, S. and Long, G., "Decentralized knowledge acquisition for mobile internet applications", ***World Wide Web***, Vol. 23, No. 5, (2020), 2653-2669, DOI: 10.1007/s11280-019-00775-w.

50. Liu, Y., Liu, Y., Liu, Z., Liang, Y., Meng, C., Zhang, J. and Zheng, Y., "Federated forest", ***IEEE Transactions on Big Data***, Vol. 10.1109/tbdata.2020.2992755, (2020), 1-1, DOI: 10.1109/tbdata.2020.2992755.

51. Li, X., Gu, Y., Dvornek, N., Staib, L.H., Ventola, P. and Duncan, J.S., "Multi-site fmri analysis using privacy-preserving federated learning and domain adaptation: Abide results", ***Medical Image Analysis***, Vol. 65, (2020), 101765, DOI: 10.1016/j.media.2020.101765.

52. Chen, S., Xue, D., Chuai, G., Yang, Q. and Liu, Q., "Fl-qsar: A federated learning-based qsar prototype for collaborative drug discovery", ***Bioinformatics***, Vol. 36, No. 22-23, (2021), 5492-5498, DOI: 10.1093/bioinformatics/btaa1006.

53. Chen, D., Xie, L.J., Kim, B., Wang, L.C., Hong, C.S., Wang, L.C. and Han, Z., "Federated learning based mobile edge computing for augmented reality applications", in 2020 International Conference on Computing, Networking and Communications, ICNC 2020, (2020).

54. Hartmann, F., Suh, S., Komarzewski, A., Smith, T.D. and Segall, I., *Federated learning for ranking browser history suggestions*, in *ArXiv*. 2019.

55. Qi, T., Wu, F., Wu, C., Huang, Y. and Xie, X., *Fedrec: Privacy-preserving news recommendation with federated learning*, in *ArXiv*. 2020.

56. Asad, M., Moustafa, A. and Ito, T., "Fedopt: Towards communication efficiency and privacy preservation in federated learning", ***Applied Sciences***, Vol. 10, No. 8, (2020), 2864, DOI: 10.3390/app10082864.

57. Preuveneers, D., Rimmer, V., Tsingenopoulos, I., Spooren, J., Joosen, W. and Ilie-Zudor, E., "Chained anomaly detection models for federated learning: An intrusion detection case study", ***Applied Sciences***, Vol. 8, No. 12, (2018), 2663, DOI: 10.3390/app8122663.

58. Brisimi, T.S., Chen, R., Mela, T., Olshevsky, A., Paschalidis, I.C. and Shi, W., "Federated learning of predictive models from federated electronic health records", ***International Journal of Medical Informatics***, Vol. 112, (2018), 59-67, DOI: 10.1016/j.ijmedinf.2018.01.007.

59. Jeong, E., Oh, S., Kim, H., Park, J., Bennis, M. and Kim, S.-L., "Communication-efficient on-device machine learning: Federated distillation and augmentation under non-iid private data", in Neural Information Processing Systems, (2018).

60. Zhao, Y., Li, M., Lai, L., Suda, N., Civin, D. and Chandra, V., *Federated learning with non-iid data*, in *ArXiv*. 2018.

61. Duan, M., Liu, D., Chen, X., Tan, Y., Ren, J., Qiao, L. and Liang, L., "Astraea: Self-balancing federated learning for improving classification accuracy of mobile deep learning applications", in Proceedings - 2019 IEEE International Conference on Computer Design, ICCD 2019, (2019), 246-254.

62. Smith, V., Chiang, C.-k., Sanjabi, M. and Talwalkar, A., "Federated multi-task learning", in Neural Information Processing Systems, (2017), 4427-4437.

63. Corinzia, L. and Buhmann, J.M., "Variational federated multi-task learning", in NeurIPS 2019 Workshop on Federated Learning for Data Privacy and Confidentiality, (2019).

64. Wang, J., Kolar, M. and Srebro, N., "Distributed multi-task learning", in Proceedings of the 19th International Conference on Artificial Intelligence and Statistics, AISTATS 2016, (2016), 751-760.

65. Sattler, F., Müller, K.-R. and Samek, W., *Clustered federated learning: Model-agnostic distributed multi-task optimization under privacy constraints*, in *ArXiv*. 2019.1-12.

66. Lim, W.Y.B., Luong, N.C., Hoang, D.T., Jiao, Y., Liang, Y.-C., Yang, Q., Niyato, D. and Miao, C., "Federated learning in mobile edge networks: A comprehensive survey", ***IEEE Communications Surveys & Tutorials***, Vol. 22, No. 3, (2020), 2031-2063, DOI: 10.1109/comst.2020.2986024.

67. Xiao, P., Cheng, S., Stankovic, V. and Vukobratovic, D., "Averaging is probably not the optimum way of aggregating parameters in federated learning", ***Entropy (Basel)***, Vol. 22, No. 3, (2020), 314, DOI: 10.3390/e22030314.

68. Li, Q., He, B. and Song, D., *Model-agnostic round-optimal federated learning via knowledge transfer*, in *ArXiv*. 2020.

69. Yurochkin, M., Agarwal, M., Ghosh, S., Greenewald, K., Hoang, T.N. and Khazaeni, Y., "Bayesian nonparametric federated learning of neural networks", in 36th International Conference on Machine Learning, ICML 2019, International Machine Learning Society (IMLS), (2019), 12583-12597.

70. Wang, H., Yurochkin, M., Sun, Y., Khazaeni, Y. and Papailiopoulos, D., *Federated learning with matched averaging*, in *ArXiv*. 2020.

71. Liu, L. and Zheng, F., *A bayesian federated learning framework with multivariate gaussian product*, in *ArXiv*. 2021.

72. Sim, K.C., Zadrazil, P. and Beaufays, F., "An investigation into on-device personalization of end-to-end automatic speech recognition models", in Proceedings of the Annual Conference of the International Speech Communication Association, INTERSPEECH, ISCA, (2019), 774-778.

73. Wang, K., Mathews, R., Kiddon, C., Eichner, H., Beaufays, F. and Ramage, D., *Federated evaluation of on-device personalization*, in *ArXiv*. 2019.

74. Schneider, J. and Vlachos, M., *Personalization of deep learning*, in *ArXiv*. 2019.

75. Mansour, Y., Mohri, M., Ro, J. and Suresh, A.T., *Three approaches for personalization with applications to federated learning*, in *ArXiv*. 2020.

76. Finn, C., Abbeel, P. and Levine, S., "Model-agnostic meta-learning for fast adaptation of deep networks", in 34th International Conference on Machine Learning, ICML 2017, (2017), 1856-1868.

77. Jiang, Y., Konečný, J., Rush, K. and Kannan, S., *Improving federated learning personalization via model agnostic meta learning*, in *ArXiv*. 2019.

78. Khodak, M., Balcan, M.-F. and Talwalkar, A., *Adaptive gradient-based meta-learning methods*, in *ArXiv*. 2019.

79. Fallah, A., Mokhtari, A. and Ozdaglar, A., *Personalized federated learning: A meta-learning approach*, in *ArXiv*. 2020.

80. Li, D. and Wang, J., "Fedmd: Heterogenous federated learning via model distillation", in NeurIPS 2019 Workshop on Federated Learning for Data Privacy and Confidentiality, (2019).

81. Ma, J., Yonetani, R. and Iqbal, Z., " Adaptive Distillation for Decentralized Learning from Heterogeneous Clients,", 2020 25th International Conference on Pattern Recognition (ICPR), 2021, pp. 7486-7492, doi: 10.1109/ICPR48806.2021.9412356.

82. Caruana, R., "Multitask learning, Learning to learn, ed. S. Thrun and L. Pratt, Boston, MA, Springer US, 10.1007/978-1-4615-5529-2_5, (1998), 95-133.

83. Liu, S., Pan, S.J. and Ho, Q., "Distributed multi-task relationship learning", in Proceedings of the ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, (2017), 937-946.

84. Ruder, S., *An overview of multi-task learning in deep neural networks*, in *ArXiv*. 2017.

85. Hinton, G., Vinyals, O. and Dean, J., *Distilling the knowledge in a neural network*, in *ArXiv*. 2015.

86. Konečný, J., McMahan, H.B., Ramage, D. and Richtárik, P., *Federated optimization: Distributed machine learning for on-device intelligence*, in *ArXiv*. 2016.

87. Sprague, M.R., Jalalirad, A., Scavuzzo, M., Capota, C., Neun, M., Do, L. and Kopp, M., "Asynchronous federated learning for geospatial applications", in Communications in Computer and Information Science, Springer Verlag, (2019), 21-28.

88. Stripelis, D. and Ambite, J.L., *Semi-synchronous federated learning*, in *ArXiv*. 2021.

89. Wu, W., He, L., Lin, W., Mao, R. and Jarvis, S., *Safa: A semi-asynchronous protocol for fast federated learning with low overhead*, in *ArXiv*. 2019.

90. Liu, Y., Yuan, X., Zhao, R., Zheng, Y. and Zheng, Y., *Rc-ssfl: Towards robust and communication-efficient semi-supervised federated learning system; rc-ssfl: Towards robust and communication-efficient semi-supervised federated learning system*, in *ArXiv*. 2018.

91. Chen, T., Jin, X., Sun, Y. and Yin, W., *Vafl: A method of vertical asynchronous federated learning*, in *ArXiv*. 2020.

92. van Dijk, M., Nguyen, N.V., Nguyen, T.N., Nguyen, L.M., Tran-Dinh, Q. and Nguyen, P.H., *Asynchronous federated learning with reduced number of rounds and with differential privacy from less aggregated gaussian noise*, in *ArXiv*. 2020.

93. Li, X., Qu, Z., Tang, B. and Lu, Z., *Stragglers are not disaster: A hybrid federated learning algorithm with delayed gradients*, in *ArXiv*. 2021.

94. Baytas, I.M., Yan, M., Jain, A.K. and Zhou, J., "Asynchronous multi-task learning", in Proceedings - IEEE International Conference on Data Mining, ICDM, (2017), 11-20.

95. Zhang, H., Li, J., Kara, K., Alistarh, D., Liu, J. and Zhang, C., "Zipml: Training linear models with end-to-end low precision, and a little bit of deep learning", in 34th International Conference on Machine Learning, ICML 2017, (2017), 6132-6140.

96. Wang, H., Sievert, S., Charles, Z., Liu, S., Wright, S. and Papailiopoulos, D., "Atomo: Communication-efficient learning via atomic sparsification", in Advances in Neural Information Processing Systems, (2018), 9850-9861.

97. Guha, N., Talwalkar, A. and Smith, V., *One-shot federated learning*, in *ArXiv*. 2019.

98. Seide, F., Fu, H., Droppo, J., Li, G. and Yu, D., "1-bit stochastic gradient descent and its application to data-parallel distributed training of speech dnns", in Proceedings of the Annual Conference of the International Speech Communication Association, INTERSPEECH, (2014), 1058-1062.

99. Wen, W., Xu, C., Yan, F., Wu, C., Wang, Y., Chen, Y. and Li, H., "Terngrad: Ternary gradients to reduce communication in distributed deep learning", in Advances in Neural Information Processing Systems, (2017), 1510-1520.

100. Strom, N., "Scalable distributed dnn training using commodity gpu cloud computing", in Proceedings of the Annual Conference of the International Speech Communication Association, INTERSPEECH, (2015), 1488-1492.

101. Aji, A.F. and Heafield, K., "Sparse communication for distributed gradient descent", in EMNLP 2017 - Conference on Empirical Methods in Natural Language Processing, Proceedings, Stroudsburg, PA, USA, Association for Computational Linguistics, (2017), 440-445.

102. Lin, Y., Han, S., Mao, H., Wang, Y. and Dally, W.J., "Deep gradient compression: Reducing the communication bandwidth for distributed training", in ICLR 2017, (2017).

103. Nishio, T. and Yonetani, R., "Client selection for federated learning with heterogeneous resources in mobile edge", in IEEE International Conference on Communications, (2019), 1-7.

104. Chen, C.Y., Choi, J., Brand, D., Agrawal, A., Zhang, W. and Gopalakrishnan, K., "Adacomp: Adaptive residual gradient compression for data-parallel distributed training", in 32nd AAAI Conference on Artificial Intelligence, AAAI 2018, (2018), 2827-2835.

105. Ahmed, A., Das, A. and Smola, A.J., "Scalable hierarchical multitask learning algorithms for conversion optimization in display advertising", in WSDM 2014 - Proceedings of the 7th ACM International Conference on Web Search and Data Mining, New York, USA, ACM Press, (2014), 153-162.

106. Mateos-Núñez, D., Cortés, J., Mateos-Núñez, D. and Cortes, J., "Distributed optimization for multi-task learning via nuclear-norm approximation∗∗the authors are with the department of mechanical and aerospace engineering, university of california,

san diego, USA", ***IFAC-PapersOnLine***, Vol. 48, No. 22, (2015), 64-69, DOI: 10.1016/j.ifacol.2015.10.308.

107. Smith, V., Forte, S., Ma, C., Takác, M., Jordan, M.I. and Jaggi, M., "Cocoa: A general framework for communication-efficient distributed optimization", ***Journal of Machine Learning Research***, Vol. 18, (2018), DOI: 10.3929/ethz-b-000282738.

108. Anil, R., Pereyra, G., Passos, A., Ormandi, R., Dahl, G.E. and Hinton, G.E., "Large scale distributed neural network training through online distillation", in 6th International Conference on Learning Representations, ICLR 2018 - Conference Track Proceedings, (2018), 1-12.

109. Tramèr, F., Zhang, F., Juels, A., Reiter, M.K. and Ristenpart, T., "Stealing machine learning models via prediction apis", in Proceedings of the 25th USENIX Security Symposium, USENIX Association, (2016), 601-618.

110. Melis, L., Song, C., De Cristofaro, E. and Shmatikov, V., "Exploiting unintended feature leakage in collaborative learning", in Proceedings - IEEE Symposium on Security and Privacy, (2019), 691-706.

111. Wei, W., Liu, L., Loper, M., Truex, S., Yu, L., Gursoy, M.E. and Wu, Y., *Adversarial examples in deep learning: Characterization and divergence*, in *ArXiv*. 2018.

112. Bagdasaryan, E., Veit, A., Hua, Y., Estrin, D. and Shmatikov, V., *How to backdoor federated learning*, in *ArXiv*. 2018.

113. Wang, H., Sreenivasan, K., Rajput, S., Vishwakarma, H., Agarwal, S., Sohn, J.-y., Lee, K. and Papailiopoulos, D., *Attack of the tails: Yes, you really can backdoor federated learning*, in *ArXiv*. 2020.

114. Biggio, B., Corona, I., Maiorca, D., Nelson, B., Šrndić, N., Laskov, P., Giacinto, G. and Roli, F., "Evasion attacks against machine learning at test time", in Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics), (2013), 387-402.

115. Abadi, M., McMahan, H.B., Chu, A., Mironov, I., Zhang, L., Goodfellow, I. and Talwar, K., "Deep learning with differential privacy", in Proceedings of the ACM Conference on Computer and Communications Security, (2016), 308-318.

116. Geyer, R.C., Klein, T. and Nabi, M., *Differentially private federated learning: A client level perspective*, in *ArXiv*. 2017.

117. Hitaj, B., Ateniese, G. and Perez-Cruz, F., "Deep models under the gan: Information leakage from collaborative deep learning", in Proceedings of the ACM Conference on Computer and Communications Security, New York, NY, USA, Association for Computing Machinery, (2017), 603-618.

118. Bogdanov, D., Laur, S. and Willemson, J., "Sharemind: A framework for fast privacy-preserving computations", in Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics), Berlin, Heidelberg, Springer Berlin Heidelberg, (2008), 192-206.

119. Araki, T., Furukawa, J., Lindell, Y., Nof, A. and Ohara, K., "High-throughput semi-honest secure three-party computation with an honest majority", in Proceedings of the ACM Conference on Computer and Communications Security, New York, NY, USA, Association for Computing Machinery, (2016), 805-817.

120. Mohassel, P. and Zhang, Y., "Secureml: A system for scalable privacy-preserving machine learning", in Proceedings - IEEE Symposium on Security and Privacy, (2017), 19-38.

121. Mohassel, P. and Rindal, P., "Aby3: A mixed protocol framework for machine learning", in Proceedings of the ACM Conference on Computer and Communications Security, New York, NY, USA, Association for Computing Machinery, (2018), 35-52.

122. Bonawitz, K., Ivanov, V., Kreuter, B., Marcedone, A., McMahan, H.B., Patel, S., Ramage, D., Segal, A. and Seth, K., "Practical secure aggregation for privacy-preserving machine learning", in Proceedings of the ACM Conference on Computer and Communications Security, New York, USA, ACM Press, (2017), 1175-1191.

123. Mandal, K. and Gong, G., "Privfl: Practical privacy-preserving federated regressions on high-dimensional data over mobile networks", in Proceedings of the ACM Conference on Computer and Communications Security, (2019).

124. Gentry, C., "Fully homomorphic encryption using ideal lattices", in Proceedings of the Annual ACM Symposium on Theory of Computing, New York, New York, USA, ACM Press, (2009), 169-178.

125. Coron, J.S., Lepoint, T. and Tibouchi, M., "Scale-invariant fully homomorphic encryption over the integers", in Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics), Springer Verlag, (2014), 311-328.

126. Brakerski, Z., Gentry, C. and Vaikuntanathan, V., "(leveled) fully homomorphic encryption without bootstrapping", in ITCS 2012 - Innovations in Theoretical Computer Science Conference, New York, New York, USA, ACM Press, (2012), 309-325.

127. Bourse, F., Minelli, M., Minihold, M. and Paillier, P., "Fast homomorphic evaluation of deep discretized neural networks", in Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics), Springer Verlag, (2018), 483-512.

128. Acar, A., Aksu, H., Uluagac, A.S. and Conti, M., "A survey on homomorphic encryption schemes", ***ACM Computing Surveys***, Vol. 51, No. 4, (2018), 1-35, DOI: 10.1145/3214303.

129. Xu, G., Li, H., Liu, S., Yang, K. and Lin, X., "Verifynet: Secure and verifiable federated learning", ***IEEE Transactions on Information Forensics and Security***, Vol. 15, No., (2020), 911-926, DOI: 10.1109/tifs.2019.2929409.

130. Itahara, S., Nishio, T., Koda, Y., Morikura, M. and Yamamoto, K., *Distillation-based semi-supervised federated learning for communication-efficient collaborative training with non-iid private data*, in *ArXiv*. 2020.

131. Van Berlo, B., Saeed, A. and Ozcelebi, T., "Towards federated unsupervised representation learning", in EdgeSys 2020 - Proceedings of the 3rd ACM International Workshop on Edge Systems, Analytics and Networking, Part of EuroSys 2020, New York, NY, USA, Association for Computing Machinery, (2020), 31-36.

132. Tian, Y., Krishnan, D. and Isola, P., "Contrastive multiview coding", in Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics), Springer Science and Business Media Deutschland GmbH, (2020), 776-794.

133. Hénaff, O.J., Srinivas, A., De Fauw, J., Razavi, A., Doersch, C., Eslami, S.M.A. and Van Den Oord, A., *Data-efficient image recognition with contrastive predictive coding*, in *ArXiv*. 2020.

134. Chen, T., Kornblith, S., Norouzi, M. and Hinton, G., *A simple framework for contrastive learning of visual representations*, in *ArXiv*. 2020.

135. Li, Q. and Song, D., *Model-contrastive federated learning*, in *ArXiv*. 2021.

136. Misra, I. and van der Maaten, L., "Self-supervised learning of pretext-invariant representations", in Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition, IEEE Computer Society, (2020), 6706-6716.

137. He, K., Fan, H., Wu, Y., Xie, S. and Girshick, R., "Momentum contrast for unsupervised visual representation learning", in Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition, IEEE Computer Society, (2020), 9726-9735.

138. Lin, J., Du, M. and Liu, J., *Free-riders in federated learning: Attacks and defenses*, in *ArXiv*. 2019.

139. Zong, B., Song, Q., Min, M.R., Cheng, W., Lumezanu, C., Cho, D.-k. and Chen, H., "Deep autoencoding gaussian mixture model for unsupervised anomaly detection", in 6th International Conference on Learning Representations, ICLR 2018 - Conference Track Proceedings, (2018).

140. Karimireddy, S.P., Kale, S., Mohri, M., Reddi, S.J., Stich, S.U. and Suresh, A.T., *Scaffold: Stochastic controlled averaging for on-device federated learning*, in *ArXiv*. 2019.

Persian Abstract

چکیده

با اختراع مدرن حسگرهای با کیفیت بالا و تراشه های هوشمند با توان محاسباتی بالا ، دستگاه های هوشمند مانند تلفن های هوشمند و دستگاه های پوشیدنی هوشمند در حال تبدیل شدن به منابع محاسباتی اصلی برای زندگی روزمره هستند. این دستگاه ها ، در مجموع ، ممکن است دارای مقدار زیادی داده ارزشمند باشند ، اما به دلیل نگرانی درباره حریم خصوصی و قوانین حریم خصوصی مانند GDPRمقررات عمومی حفاظت از داده ها، این مقدار عظیم داده های بسیار ارزشمند برای آموزش مدل ها برای کاربردهای دقیق تر و کارآمد هوش مصنوعی در دسترس نیست. . آموزش فدراسیون (FL)به عنوان یک روش یادگیری مشارکتی بسیار برجسته برای یادگیری از چنین داده های خصوصی غیرمتمرکز ظهور کرده و در عین حال محدودیت های حریم خصوصی را نیز برآورده می کند. برای یادگیری از چنین داده های غیرمتمرکز و توزیع شده گسترده ، یادگیری فدرال باید بر برخی چالش های منحصر به فرد مانند ناهمگنی سیستم ، ناهمگنی آماری ، ارتباطات ، ناهمگنی مدل ، حریم خصوصی و امنیت غلبه کند. در این مقاله ، برای شروع ، برخی از اصول یادگیری فدراسیون به همراه تعریف و کاربردهای FL را توضیح می دهیم. پس از آن ، ما چالش های منحصر به فرد FL را بیشتر توضیح می دهیم در حالی که رویکردهای اخیراً پیشنهادی برای کنترل آنها را به طور انتقادی پوشش می دهیم. علاوه بر این ، این مقاله همچنین برخی از چالش های نسبتاً جدید برای یادگیری فدراسیون را مورد بحث قرار می دهد. برای نتیجه گیری ، ما در مورد برخی از مسیرهای تحقیقاتی آینده در حوزه یادگیری فدراسیون بحث می کنیم.