IEEE Access
Multidisciplinary | Rapid Review | Open Access Journal

Received July 7, 2020, accepted July 25, 2020, date of publication July 31, 2020, date of current version August 12, 2020.

Digital Object Identifier 10.1109/ACCESS.2020.3013541

Federated Learning: A Survey on Enabling Technologies, Protocols, and Applications

MOHAMMED ALEDHARI[1], (Member, IEEE), REHMA RAZZAK[1], REZA M. PARIZI[1], (Senior Member, IEEE), AND FAHAD SAEED[2], (Senior Member, IEEE)

[1]College of Computing and Software Engineering, Kennesaw State University, Marietta, GA 30060, USA
[2]School of Computing and Information Sciences, Florida International University, Miami, FL 33199, USA

Corresponding author: Reza M. Parizi (rparizi1@kennesaw.edu)

This work was supported in part by the National Institute of General Medical Sciences (NIGMS) of the National Institutes of Health (NIH) under Award R01GM134384, and in part by the National Science Foundation (NSF) CAREER under Grant OAC 1925960.

ABSTRACT This paper provides a comprehensive study of Federated Learning (FL) with an emphasis on enabling software and hardware platforms, protocols, real-life applications and use-cases. FL can be applicable to multiple domains but applying it to different industries has its own set of obstacles. FL is known as collaborative learning, where algorithm(s) get trained across multiple devices or servers with decentralized data samples without having to exchange the actual data. This approach is radically different from other more established techniques such as getting the data samples uploaded to servers or having data in some form of distributed infrastructure. FL on the other hand generates more robust models without sharing data, leading to privacy-preserved solutions with higher security and access privileges to data. This paper starts by providing an overview of FL. Then, it gives an overview of technical details that pertain to FL enabling technologies, protocols, and applications. Compared to other survey papers in the field, our objective is to provide a more thorough summary of the most relevant protocols, platforms, and real-life use-cases of FL to enable data scientists to build better privacy-preserving solutions for industries in critical need of FL. We also provide an overview of key challenges presented in the recent literature and provide a summary of related research work. Moreover, we explore both the challenges and advantages of FL and present detailed service use-cases to illustrate how different architectures and protocols that use FL can fit together to deliver desired results.

INDEX TERMS Federated learning, machine learning, collaborative AI, privacy, security, decentralized data, on-device AI, peer-to-peer network.

I. INTRODUCTION

Federated Learning (FL) is a newly introduced technology [1] that has attracted a lot of attention from researchers to explore its potential and applicability [2], [3]. FL simply attempts to answer this main question [4], *can we train the model without needing to transfer data over to a central location?* Within the FL framework, the focus is geared towards collaboration, which is not always achieved through standard machine learning algorithms [5]. In addition, FL allows the algorithm(s) used to gain experience, which is also something that cannot always be guaranteed through traditional machine learning methods [6], [7]. FL has been employed in a variety of applications, ranging from medical to IoT, transportation, defense, and mobile apps. Its applicability makes FL highly reliable, with several experiments having been conducted already. Despite FL's promising potential, FL is still not widely understood in regard to some of its technical components such as platforms, hardware, software, and others regarding data privacy and data access [8], [9]. Therefore, our focus in this paper is to expand on FL's technical aspects while presenting detailed examples of FL-based [10] architectures that can be adapted for any industry.

Because of strict regulations regarding data privacy, it is usually considered not practical to gather and share consumers' data within a centralized location. This also challenges traditional machine learning algorithms because they require huge quantities of data training examples to

The associate editor coordinating the review of this manuscript and approving it for publication was Shiping Wen.

learn [11]. The reason for traditional machine learning algorithms having these caveats is due to how their models get trained [12], [13]. Traditional machine learning usually has a main server that handles data storage and training models. Typically, there are two ways of using these trained models of machine learning [14]. Either we build a pipeline for the data so it can pass through the server, or transfer the machine learning models to any device that interacts with the environment [15]. Unfortunately, both of these approaches are not optimal because their models are not able to rapidly adapt.

With FL, the models get trained at the device level. So the models are brought over to the data sources or devices for training and prediction [16]. The models (i.e., models' updates) are sent back to the main server for aggregating. Then, one consolidated model gets transferred back to the devices using concepts from distributed computing [17]. This is so that we can track and re-distribute each of the models at various devices. FL's approach is very advantageous for utilizing low-costing machine learning models on devices such as cell phones and sensors [18]. A figure representing the general architecture of FL is shown in Fig. 1

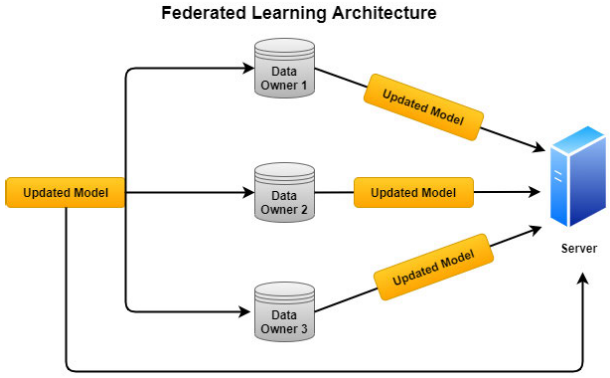

FIGURE 1. General federated learning architecture.

FL has unique use cases, with plenty of research relating to FL's applications, one example being in the healthcare sector [19], [20]. Fig. 2 illustrates how one would apply an FL architecture in a healthcare setting. Unfortunately, there are still some crucial obstacles for FL to be fully incorporated in other settings, especially regarding the data. For our models and algorithms to learn effectively to obtain optimal results, it requires a lot of data in order to ensure our models will be as accurate as possible [21]. However, even the data itself can prove difficult to handle because there is often a lot of diversity within the data, such as contents, structure, and file-formats. Additionally, the idea of centralizing streams of sensitive data over to tech companies has proven to be very unpopular with the United States [22], [23]. According to MIT computer science associate professor Ramesh Raskar, the dichotomy between data privacy and the benefits of using that data on society is false. The reasoning is that we can

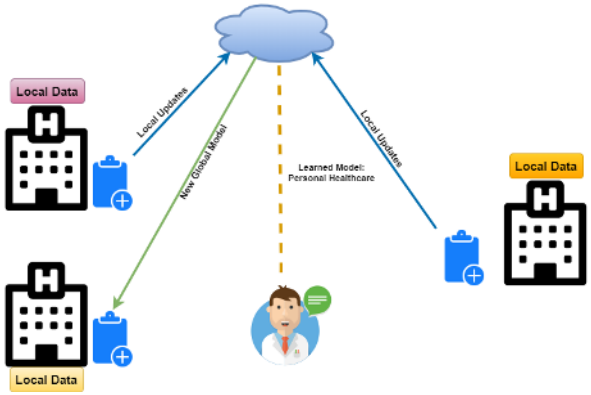

FIGURE 2. Federated learning architecture applied in a hospital setting.

achieve both utility plus privacy, so concerns about privacy can be lessened [24].

A. PURPOSE OF STUDY

This work's purpose is to expand on the current platforms, protocols, and architectures of FL. While there is research on this topic, sufficient progress has not been made in regard to understanding FL on a deeper, technical level. Not only is FL still new, but it is also not widely understood and there has been little application of it in most industries. So, because of this, we currently do not have a sufficient understanding of FL nor are we able to see the bigger picture of how exactly FL can benefit multiple industries. Specifically, this work will attempt to answer the questions below:

- What are current architectures and platforms that are used for Federated Learning?
- What are the current hardware and software technologies that are used in Federated Learning?

In answering these questions, we hope this work can allow for FL to be applied towards more industries. That way we gain a comprehensive picture of how FL directly impacts these industries.

B. CONTRIBUTIONS

The work hopes to contribute a comprehensive overview of FL in terms of definition, applicability, and usefulness. Several works have studied this topic, but how our work stands out in comparison to previous works is that we manage to dive deep into the architectures of FL and its use cases. In doing so, the work can contribute an overall blueprint for data scientists and other researches on designing FL-based solutions for alleviating future challenges. So, this work contributes to the following:

- Compared to other survey papers on FL, this survey provides a deeper summary of the most relevant FL hardware, software, platforms, and protocols to enable researchers to get up to speed about FL quickly, giving them enough knowledge to pursue the topic of Federated Learning without having to endure possible steep learning curves.

- We also provide solid examples of applications and use cases of FL to illustrate how different architectures of FL can be applied for multiple scenarios, allowing the audience to better understand how FL can be applicable. Besides, highlighting use cases and applications of FL in particular medical settings would allow healthcare professionals to have more faith in streamlining their data for FL.
- We provide an overview of some of the key FL challenges presented in the recent literature and provide a summary of related research work. Additionally, we offer insight into best design practices for designing FL models.

C. PAPER ORGANIZATION

There are twelve sections within this paper. Section II discusses related works that have studied FL. Section III goes over architectures and platforms that are used for FL. Section IV discusses the enabling technologies of FL, such as hardware/software, and algorithms geared toward FL. Section V discusses what optimization techniques can be applied to some FL models while Section VI discusses various network protocols that can support FL. Section VII discusses the limitations and challenges of FL. Then, Section VIII discusses the market implications of FL. Section IX discusses the benefits and costs of FL and Section X discusses in detail the applications and use-cases of FL. Section XI goes over the best practices in designing FL models and finally Section XII concludes the paper.

II. RELATED WORKS

Most notably, FL is often compared against distributed learning, parallel learning, and deep learning. While FL remains a new topic, there have been several related works that do examine FL in detail. Table 1 summarizes various works that tackle FL, along with other topics focusing on use-cases for FL.

TABLE 1. Summary of related works on FL.

Ref. No	Author (s)	Article Topic(s)
[24]	Y. Xia	Deep Learning
[25]	Tal Ben-Nun, T. Hoefler	
[26]	M.G. Poirot, et al.	
[27]	P. Vepakomma, et al.	HIPAA Guidelines for FL
[28]		Drawbacks of FL
[29]	Kevin Hsieh	Traditional ML Methods
[30]	Qinbin Li, et al.	Data Privacy and Protection
		Future Direction of FL
		Challenges of FL
[31]	V. Kulkarni, et al.	Personalization techniques for FL
[32]	J. Geiping, et al.	Privacy of FL
[33]	Y.Liu, et al.	FL for 6G

A. DEEP LEARNING

In comparison to our work, articles [25]–[28], and [29] only cover Deep Learning and its comparisons to FL. FL is often compared to Deep Learning techniques because DNNs (Deep Neural Networks) have been used for various purposes and often have promising results. Unfortunately, DNNs are also prone to several drawbacks which make it more difficult to incorporate into FL. DNNs are not always optimal for FL because as data-sets increase, so do the DNNs' complexity. So the DNN's complexity is proportional to the computational requirements and memory demands. Additionally, Deep Learning models applied for FL need to be able to still work without accessing raw client data, so privacy is the main focus. Comparisons of deep learning models have been made in regard to the protection offered, performances, resources, etc. Both of these aspects depend on architecture, which is something FL also needs to be successful. In addition to considering these metrics, we also need to consider the HIPAA guidelines because data cannot be shared with external entities. Overall, the authors do an excellent job of diving into deep learning, but they do not dive into enough detail about FL.

B. DRAWBACKS OF USING MACHINE LEARNING

The article by [30] discusses how cost and latency still plague traditional machine learning (ML) methods. Both of these problems are difficult to solve because the data itself is highly distributed. This poses problems with traditional machine learning methodologies in regards to computation and communication. For instance, if the data spans multiple locations, data centers' communication can very easily overwhelm the limited bandwidth. While Machine Learning does allow us to pre-process data to reduce latency, it can still lead to high monetary costs. For communication, because the data can be highly independent and highly distributed, this can create some inter-operability issues with Machine Learning algorithms using the data, because the data could end up not being useable. While the article does highlight the drawbacks of traditional machine learning methods, that is not the primary focus of our paper.

C. FEDERATED LEARNING

Another work by [31] talks about FL in regards to future direction, and feasibility for data privacy and protection. They also compare various FL systems. FL systems face challenges such as system assumptions being unpractical, efficiency, and scalability. It should be noted that when designing FL systems, it seems that two main characteristics are rarely considered in implementation, which is *heterogeneity* and *autonomy*.

- **Heterogeneity** refers to diversity, systems that use more than one kind of processors or cores (this is known as Heterogeneous Computing) to achieve optimal performance and energy efficiency. In the context of Machine Learning and FL, Heterogeneity concerns data, privacy, and task requirements.

- **Autonomy** refers to independent control. The systems involved in FL need to be willing to share information with others. The authors break down autonomy into several different categories, such as association and communication autonomy. Both of them highlight the ability to associate with FL, the ability to partake in more than one FL system, and deciding how much information should be communicated to others.

The authors also manage to classify FL Systems by a few aspects: distributing data, the model, privacy frameworks, communication architectures, etc. The work also manages to expand on the concept of FL by explaining two different types of FL systems: *private and public*.

- **Private FL systems** have few amounts of entities, each with huge amounts of data and computation power. The challenge that private FL systems have to overcome is how to transfer computations to data centers under certain constraints.
- **Public FL Systems** have a large number of entities but they have small quantities of data and computation power.

The authors manage to examine other related works on FL, making this work a reliable foundation of FL. However, the article could be stronger in providing solid examples of use-cases and real-life applications to remedy some of the challenges mentioned. Our work contains plenty of use-cases and applications to provide a comprehensive overview of how FL can remedy the challenges mentioned in the article.

In addition, the authors V. Kulkarni, M. Kulkarni, and A. Pant conduct a survey of techniques for FL. They focus specifically on personalization techniques. Here, the authors mention why we need to consider personalization. Personalizing a global model is necessary in order to account for FL's challenge of heterogeneity. Most techniques for personalizing a model usually require two steps; In the first step, a global model is built. In the second step, the global model is modified for each client via private data from that client. The article also mentions other techniques for building global models, some of which include transfer learning, multi-task learning, meta-learning, and others [32]. The article does well in distinguishing these techniques from each other. Unfortunately, the article discusses FL regarding one challenge of it. Additionally, there are no use-cases or examples presented regarding the techniques the article mentions. What is also missing is the hardware/software platforms for such techniques. As such, it seems only a surface level of FL is covered.

Another work that dives into the topic of FL is [33]. In their work, the authors focus on parameter updates for FL, which are crucial to ensure the models are working with the most recent parameters. The article attempts to answer the question of how easy is it to break privacy in FL, which is an essential question to answer in order to effectively implement FL. The article not only discusses FL in detail but discusses in detail the privacy limitations of FL both in theory and practice. Interestingly, the article focuses on using FL to recover image data, which is an interesting use case that could be beneficial to the community. Another aspect of FL that is effectively discussed is the impact of both the network architecture and state of parameters. Admittedly, their work is more implementation-focused and contains quite a bit of math that may be difficult for newcomers in FL to follow.

The article [34] discusses FL's challenges, methods, and future directions in regard to 6G communications. While this is beyond the scope of our paper, the article effectively highlights both the drawbacks of using traditional machine learning for 6G and the potential of FL to improve the likelihood of 6G communications. The article emphasizes several challenges of FL regarding this, such as expensive communication cost, security, and privacy. The authors effectively dive into the detail of each of the aforementioned challenges, while presenting an architecture of what FL could like in 6G.

III. ARCHITECTURES AND PLATFORMS OF FEDERATED LEARNING

FL comes with quite a few architectures and platforms. In regards to the medical field, there are already several institutions that are trying to develop FL architectures [35], [36]. One of the leading institutions is the University of Pennsylvania and Intel. Besides, many platforms have also been developed for FL, and some will be discussed in this section. Table 2 summarizes the various architectures and their focuses. These architectures are talked about in more detail in this section.

TABLE 2. Summary of architectures and their focus.

Architecture	Benefit(s)	Focus
Horizontal FL	Independence	Security
Vertical FL	Encryption	Privacy
FTL	Higher accuracy Encryption	Prevents accuracy loss
MMVFL	Label sharing Multiple participants	Complex scenarios Data leakage
FEDF	Improved training speed	Parallel training Privacy preservation
PerFit	Cloud-based Local-sharing	IoT applicability
FedHealth	Powerful models More generalization More applicability	Healthcare
FADL	No data aggregation needed Reduced classification errors	EHRs
Blockchain-FL	High efficiency Enhanced security	Industrial IoT

According to the author(s) of [37], the architecture that we use for FL is dependent on data distribution, specifically the distribution characteristics of the data. The authors define two types of FL architectures: Horizontal FL and Vertical FL. These two architectures are different in terms of how their architectures are structured, and their definitions. Horizontal

FL(also known as sample-based FL), is where features are similar but vary in terms of data. Interestingly, there have been examples of a Horizontal FL framework being proposed. One example is where Google proposed a Horizontal FL approach for managing the android mobile phone updates. From a more technical perspective, Horizontal FL assumes there are honest consumers and security against a server. So just the central server can modify consumers' data [35]. With Horizontal FL's architecture, there are an x amount of elements of similar structures that learn a model with the aid servers or parameters; the Training process of Horizontal FL is shown in Fig. 4. Vertical FL is also known as feature-based FL; Fig. 3 illustrates the workflow of Vertical FL. Here, data-sets can have similar sample IDs but differ in their features. With Vertical FL, what we are doing is collecting and grouping these various features. Then we need to calculate training loss so we can form a model that contains data from both entities collaboratively. Under Vertical FL, each entity has the same identity and status. With regard to security, the Vertical FL system also assumes there are honest consumers. However, the security aspect of Vertical FL concerns two entities. Also, in this framework, the adversary (assuming one of these two entities is compromised) can only learn data from the compromised client. With the Vertical FL architecture, this architecture is less explored than the Horizontal FL architecture. There are two main parts in the Vertical FL architecture: (1) Encrypted entity alignment and (2) Encrypted model training [35], [37]. A benefit of this architecture is that it is independent of other machine learning methods. Interestingly, Horizontal FL has been used in medical cases, such as drug detection.

1) The participants locally calculate gradients. Then after computation, they need to mask that selection of gradients via encryption, privacy techniques, or secret sharing techniques. The results are transferred to servers.
2) Servers handles the collection and does so without learning anything from the participant(s)
3) Server sends the results back to the participants.
4) Participants update the model accordingly.

In addition to the Horizontal FL and Vertical Architectures, another architecture for FL is called Federated Transfer Learning (FTL). FTL was proposed in [38]. With FTL, it is used to utilize data from a different source for training the model(s). With transfer learning, it involves learning a common representation between one entity's features and reducing error in predicting the labels for the targeted entity. That way, accuracy loss is minimal. FTL obtained huge attention in various industries, especially healthcare [39]. To avoid potentially exposing client data, FTL utilizes encryption and approximation to make sure privacy is in fact, protected. So as a result, the actual raw data and models are both kept locally [40]. There are also three components of the FTL system:

1) **Guest** - Data holder. Guests are responsible for launching task-based and multi-party model training with data-sets provided by both itself and the Host. They mainly deal with data encrypting and computation.
2) **Host** - Also a Data holder.
3) **Arbiter** - Sends public keys to both the Guest and Host. The Arbiter is primarily responsible for collecting gradients and check whether the loss is converging.

Regarding the workflow of FTL, it is summarized in Fig. 5. With the workflow, the Guest and Host first, need to compute and encrypt their results locally via their data. The data is used for gradient and loss computations. Then they transfer the encrypted values to Arbiter. Afterward, the Guest and Host obtain the gradients and loss computations from the Arbiter to modify the models. The FTL framework continues iteratively

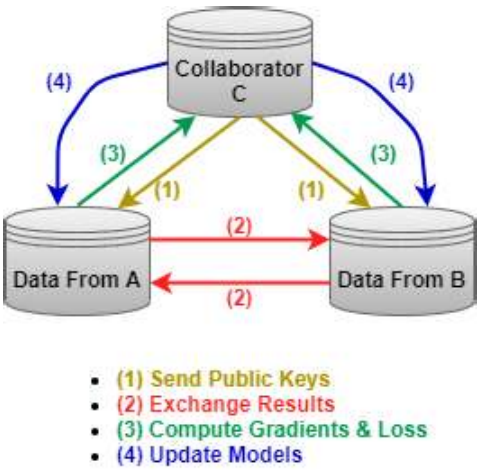

FIGURE 3. Vertical federated learning arcitecture.

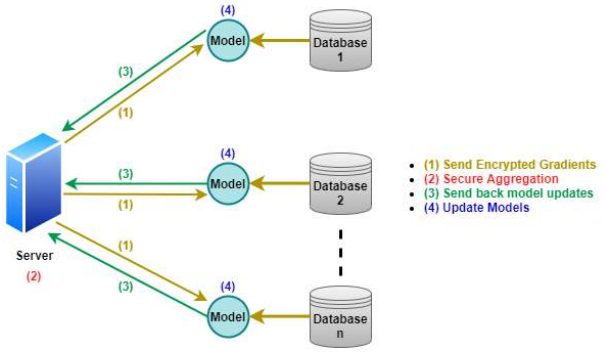

FIGURE 4. Horizontal federated learning architecture.

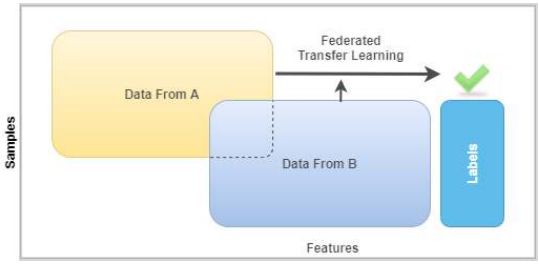

FIGURE 5. Framework of federated transfer learning.

until the loss function converges [38]. FTL also supports two different training approaches: *heterogenous* and *homogenous*. The homogenous approach is where entities help train the model(s) with differing kinds of samples. Heterogenous is where entities share the same samples but in different feature spaces. They group these features in an encrypted state and then build a model with all the data collaboratively. Regarding the performance of FTL, FTL was meant to replace deep-learning approaches since deep-learning approaches are prone to accuracy loss. The authors that proposed FTL were able to conclude that FTL's accuracy is higher. The authors of the proposed framework were able to conclude that FTL is more scalable and flexible. Unfortunately, FTL does have some limitations. For instance, it needs entities to exchange encrypted results from only the common representation layers. So they do not apply to all transfer mechanisms [38], [39].

Another study by Siwei Feng and Han Yu proposes a new architecture based on the Vertical FL system. Specifically, the authors' proposed architecture is called the Multi-Participant Multi-class Vertical Federated Learning framework (MMVFL). This particular framework, shown in Fig. 6, is supposed to handle multiple participants. The authors note that MMVFL enables label sharing from its owner to other participants in a manner of privacy preservation. One problem with using Horizontal FL architecture is the assumption that data-sets from different entities have the same feature area yet they may not similar to the same sample ID space. That is not always the case, unfortunately, so the proposed framework is meant to alleviate that setback. With the MMVFL framework, the goal is to learn multiple frameworks to complete different various objectives. The reason for doing so is to make the learning process more personalized. To evaluate the framework's performance, the authors used two computer-vision data-sets. The authors also compare their framework against other methods. Their framework achieved better results depending on how many features used. The MMVFL framework was also noted to perform better by using a smaller amount of features as well [41].

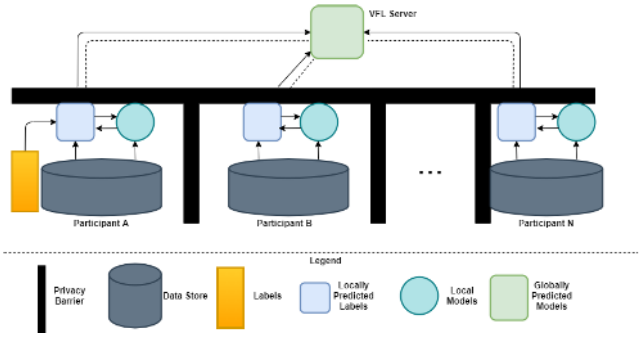

FIGURE 6. Framework of MMVFL.

Another framework of FL is proposed by Tien-Dung *et al.* [42]. Their framework in terms of FL is geared toward privacy preservation and parallel training. Their framework, called FEDF, allows a model to be learned on multiple geographically distributed training data-sets which could belong to different owners. The authors' proposed architecture consists of a master and X amount of workers as shown in Fig. 7. The master is responsible for handling the training process, which consists of the following:

- Initializing execution of the training algorithm
- Obtaining all the training outcomes from the workers
- Modifying the global model instance(s)
- Notifying the workers after the global model instance has been modified and ready for the next training session.

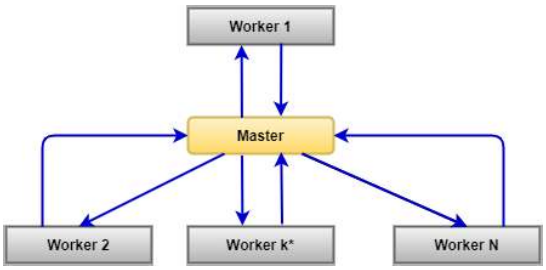

FIGURE 7. FEDF framework.

Meanwhile, the workers represent the data owners that are equipped with a computing server to run a training model on their data. When having a model instance downloaded from the master, each worker runs the training method. Then they send over training outcomes to the master by using the communication protocol that is set up. We make the assumption that each worker is responsible for determining what parameters are used in the training algorithm such as the learning rate. These parameters are the private information of each worker and they are kept unknown to the master as well as other workers. So it prevents others from gaining information about the training data. Something to keep in mind is that the size of the workers' data-sets is not homogeneous. So with this in mind, training time will vary between workers. The authors manage to implement their framework using Python and TensorFlow. To handle communication between the workers and master, the authors used secure socket programming. The authors managed to test their framework on a variety of machines as well. The FEDF framework was tested on different datasets, mainly the CIFAR-10 membrane data-set named MEMBRANE, and a medical image data-set called HEART-VESSEL. The evaluation metrics used were training speed, performance, and amount of data exchanged. Upon experimentation, the authors were able to conclude their proposed framework was able to improve training speed nine times faster without sacrificing accuracy.

Another interesting framework of FL comes from Qiong Wu *et al.* [43]. Their proposed architecture for FL centers around IoT applicability. While IoT is not the focus of this paper, It is noteworthy to mention that FL has been proposed for IoT. [44], [45]. The authors' proposed framework is called

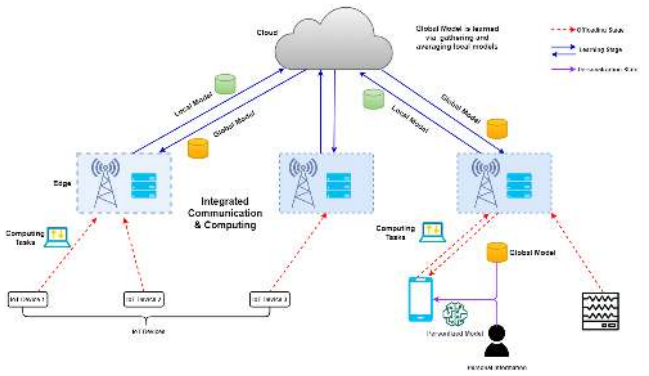

FIGURE 8. Framework of PerFit.

PerFit, shown in Fig. 8. PerFit was meant to alleviate some challenges of both FL and IoT. These challenges include:

- **Heterogeneity of Devices** - Devices that are used in the medical community and in general are all different in terms of hardware, such as CPU, memory, networking bandwidth, capacity, and power. This could create high communication costs when applying FL, so extra factors need to be considered, such as fault tolerance. Additionally, medical devices may end up dropping out of the various learning processes due to bad connectivity and energy constraints.
- **Statistical Heterogeneity**- This deals with different usage scenarios and settings. In the context of healthcare, the distributions of users? activity data can vary by a lot depending on the users? diverse physical characteristics and behavioral habits. Another thing to consider is that data samples across devices could vary significantly.
- **Heterogeneity of Models** - If we were to examine the original FL framework, we would see that the involved devices would need to agree on a specific architecture of the training model. In doing so, we can obtain an overall effective model by collecting all the model weights that were collected from the local models. However, this approach would likely need to be modified for medical communities [46]. Different devices want to form their own models based on their environment and resource capacities. Due to privacy concerns in healthcare, sharing of model specificities would be much harder.

Examining PerFit's architecture, PerFit's architecture is cloud-based, which the authors say will bring readily available computing power for IoT devices. The architecture structured so that each IoT device can unload its computing tasks so that efficiency requirements and low latency requirements can be fulfilled. FL is applied towards devices, servers, and the cloud. In doing so, models can be shared locally without compromising sensitive data. The learning process of the PerFit Framework has three stages:

1) **Unload Stage** - Here in this stage, the IoT device can transfer its learning model and data samples to the cloud for fast calculation.
2) **The Learning Stage** - In this step, the device and the cloud both compute models depending on samples of data, then transmit information. Then the server collects information submitted, then averages them into a global model. The model information exchange process repeats until it converges after a certain amount of iterations. The end result is an optimal global model that can be modified for further personalization.
3) **The Personalization Stage** - In this last stage, each device trains a personalized model in order to capture specific characteristics and requirements. This is based on the global model's information and its own personal intel. The specific learning operations at this stage depend on the adopted federated learning mechanism.

To verify PerFit's effectiveness, the authors used a data-set called Mobile-Act, which centers on human activity recognition. The data-set has ten kinds of activities such as walking, falls, jumping, jogging, etc. The authors actually compared two different FL approaches as well: FTL and Federated Distillation (FD) in terms of performance. The authors' result confirmed that their proposed framework, PerFit was effectively and can be considered a promising framework for future FL implementations.

While there are plenty of architectures for FL in general, there do not seem to be many FL-architectures geared towards specific industries [47]. However, this does not mean such architectures do not exist; There are simply fewer FL-frameworks that center around specific industries. Technically, the FL architectures discussed so far could potentially work for any industry, but there would likely be some hurdles in modifying the mentioned FL frameworks to fit them [48]. In regards to the system architecture, we need to take into account these possible challenges:

- Ensuring data integrity when communicating
- Designing secure encryption methods that take full advantage of the computational resources
- Use devices to reduce idle time

One example of an FL architecture geared towards a specific industry comes from [49]. In the article, the authors propose an FL framework centering around healthcare, specifically wearable devices, which fall into the category of Smart Healthcare. The proposed architecture is called FedHealth, shown in Fig. 9. According to the authors, the challenges that Smart Healthcare is faced with are a lack of

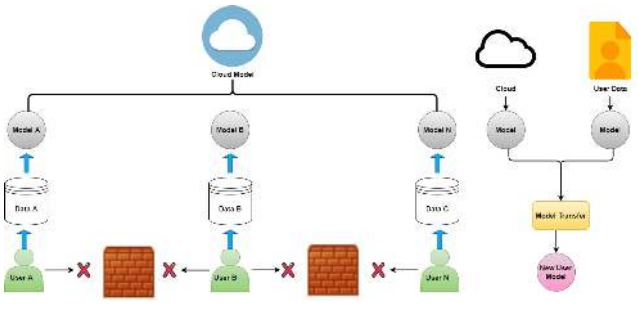

FIGURE 9. FedHealth framework.

personalization and how to access user data without violating privacy. The users' data is often very isolated, making it difficult to aggregate. The authors claim their proposed architecture is the first FL framework to address these challenges. What FedHealth does is gather information from individual institutions to generate robust frameworks without compromising the users' privacy. After the model is made, FedHealth uses other methods for obtaining well-tailored models for individual institutions. Simply speaking, transfer learning involves transmitting information via current entities to a new entity. FedHealth assumes there are three institutions and one server, which can be extended to be more generalized. There are four procedures in the framework.

Interestingly, FedHealth architecture uses Deep Neural Networks (DNNs) for learning both models. DNNs are used in the framework to learn features and train classifiers. FedHealth works continuously with new emerging user data. The framework can update both models at the same time when facing new user data. So the longer the consumer uses the product, the more personalized the model can be. Other than transfer learning, FedHealth can also incorporate other popular methods for personalization. Additionally, the authors' proposed framework allows them to adopt other conventional machine learning methods, making the framework more generalized and applicable. The authors test their framework by using a data-set centered on human activity recognition. The data-set, called UCI Smartphone contained six activities collected from 30 users. The six activities were walking, walking upstairs, walking downstairs, sitting, standing, and laying. The authors split the data-set with a 70-30 Train-Test ratio and tested against a few traditional machine learning algorithms. The FedHealth framework managed to out-perform all of them. The framework itself is very applicable to the healthcare setting.

Another architecture geared towards the medical field comes from the authors Dianbo Liu, Timothy Miller, *et al.* Their proposed architecture is called Federated-Autonomous Deep Learning (FADL), shown in Fig. 10. This architecture dealt with Electronic Health Records (EHR). EHR data is usually collected via individual institutions and stored across locations. Unfortunately, obtaining access to EHR data can be difficult and slow thanks to security, privacy, regulatory, and operational setbacks. According to the authors, the FADL framework trains some parts of a model using all data sources plus additional parts using data via certain data resources. To test their framework, the authors used ICU hospital data. The authors also used data from fifty-eight hospitals. Regarding FADL's structure, the structure consists of an artificial neural network (ANN) with three layers. Upon testing, the authors were able to conclude that FADL managed to out-perform traditional FL [50].

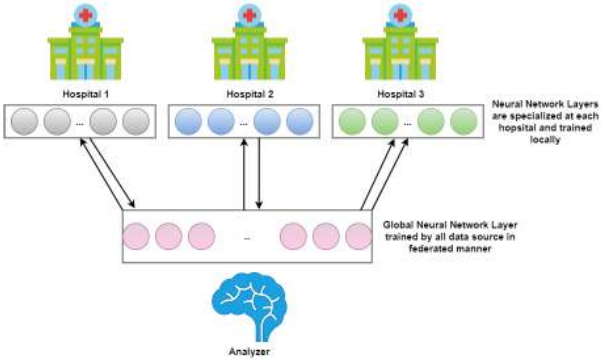

FIGURE 10. Framework of FADL.

Furthermore, another architecture that uses FL and is geared towards a specific industry comes from Yunlong Lu *et al.* Here, the authors propose an FL architecture for blockchain. Specifically, the industry targeted for this FL-blockchain based framework is the Industrial IoT. The architecture is meant to solve data-leakage problems relating to the security and privacy issues in FL. The proposed architecture has two modules: a blockchain module and a federated learning module. The blockchain module establishes secure connections among all the IoT devices via encrypted records, which is maintained by entities equipped with computing and storage resources, such as base stations. There are two types of transactions in the blockchain module: retrieval transactions and data sharing transactions. For privacy concerns and storage limitations, the blockchain is only used for obtaining related data and handling data accessibility [51]. Fig. 11 illustrates a working mechanism of the proposed architecture. The authors were able to evaluate their proposed architecture, and results indicated that the proposed architecture achieved great accuracy, high efficiency, and enhanced security.

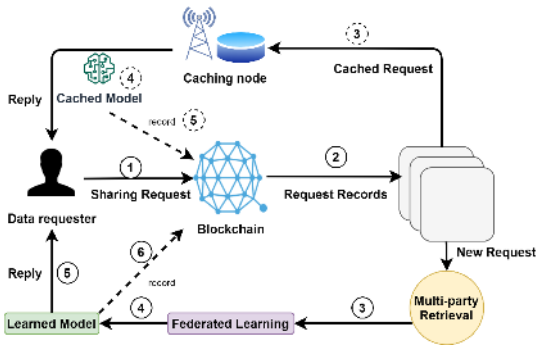

FIGURE 11. FL-based framework that incorporates blockchain.

Several platforms exist for FL as well. Thanks to the growth of FL, there are a lot of industries and research teams that research FL for product and research development [52]. There are several popular platforms for FL, which have been summarized in Table 3 in terms of their focus and supporting software. There are plenty of platforms and architectures for FL, with more examples including [53]–[56]. These platforms and architectures help refine FL better. One other architecture involves using blockchain on FL, where the authors of [57]–[59], and [60] propose an architecture incorporating both Blockchain and FL. By proposing this architecture, there

TABLE 3. A Summary of FL platforms, their focus and supporting software.

Platform	Focus	Supporting software
PySyft	Privacy	Python library
FATE	FL	FATEBoard FATEFlow
Tensor\IO	Mobile Devices	Tensor Flow
FFL-ERL	Parallel computing Real-time Systems	Erlang
CrypTen	Preserving Privacy	PyTorch
LEAF	Multi-tasking	Python library
TFF	FL	Tensor Flow

will be no need for centralized training data, and devices could potentially get trained much faster. The architectures discussed is not only applicable for FL in general, but it can apply for other industries too.

- **PySyft** - PySyft is mainly geared towards privacy. It handles private data from the models' training using federated learning within PyTorch, which is another library in the Python library.
- **Tensor Flow Federated (TFF)** - TFF is another platform for federated learning. TFF, it provides users with a more flexible and open framework for their needs.
- **FATE (Federated AI Technology Enabler)** - FATE is another open-source project geared towards FL. The platform was initiated by the Webank AI division. The goal of FATE is to provide a secured computing architecture, where a secure computing protocol is implemented based on encryption. FATE can support various federated learning architectures and machine learning algorithms, including logistic regression, transfer learning, etc. Through multiple upgrades, the platform was able to have a new tool that allowed for a more visual approach to FL: FATEBoard. The new upgrades also let the platform have FL modeling pipeline scheduling and life cycle management tools called FATEFlow.
- **Tensor/IO** - This particular platform is a platform that brings the power of TensorFlow to mobile devices such as iOS, Android, and React native applications. While this platform does not implement any machine learning methods, the platform cooperates with TensorFlow to ease the process of implementing and deploying models on mobile phones. It can run on iOS and Android phones. The library also has choices of back-end programming languages the consumer can choose from when using this platform. The languages it supports are objective-c, Swift, Java, Kotlin, or JavaScript. A benefit of using this platform is that prediction can be done in as little as five lines of code.
- **Functional Federated Learning in Erlang (FFL-ERL)** - Erlang is a structured, dynamic-typed programming language that has built-in parallel computing support, which is suitable for establishing real-time systems. This particular platform was proposed by the authors Gregor Ulm, Emil Gustavsson, and Mats Jirstran back in 2018. In the authors' work, they go through the mathematical implementation of their proposed platform. For testing, the authors actually generated an artificial data-set. This particular platform, unfortunately, does have a performance penalty [61].
- **CrypTen** - CrypTen is actually a framework geared towards privacy preservation built on PyTorch. There are a few benefits to using this platform, especially for machine learning. One benefit is that the platform was made with real-world challenges in mind. So it has the potential to be applicable to a lot of real-world medical care challenges. Another benefit is that CrypTen is library-based. So it is easier for users to debug, experiment on, and explore different models. Currently, the CrypTen platform runs only on the Linux and Mac operating systems.
- **LEAF** - LEAF is a framework for FL, multi-tasking, meta-learning, etc. This platform has several data-sets available for experimentation. The authors of [62] proposed the framework LEAF back in 2019. Core components of LEAF consists of three components: the datasets, implementation references, and metrics.

IV. ENABLING TECHNOLOGIES: USED HARDWARE, SOFTWARE, AND ALGORITHMS

There is plenty of hardware, software, and algorithms that have been applied toward FL. These technologies have been utilized to further refine FL. These technologies can also be applied to different communities too.

A study by [63] manages to evaluate and compare several algorithms for FL. The mentioned algorithms include FedAvg, and Federated Stochastic Variance Reduced Gradient (FSVGR), and CO-OP.

FedAvg is one of the algorithms for FL. The way the FedAvg algorithm works is that it initializes training thanks to the main server, where the hosts share an overall global model [63]. Optimization is done via the Stochastic Gradient Descent Algorithm (SGD). In addition, the FedAvg algorithm has five parameters that need to be accounted for: the number of clients, batch sizes, amount of epochs, learning rate, and decay. The FedAvg algorithm begins by starting up the global model. The server chooses a group of clients and transmits the recent model to all the clients. Then After modifying the local models to the shared model, clients divide their own data into different batch sizes and perform a certain amount of epochs of SGD. Then the clients transfer their newly modified local models to the server. The server makes new global models and does so by calculating a weighted total of all the obtained local models [63].

Unfortunately, the FedAvg algorithm is not without a few setbacks. While it has been proven to be successful, the FedAvg algorithm still does not tackle all the challenges associated with heterogeneity. Specifically, FedAvg does not

allow the devices involved to perform various amounts of local work based on the constraints of their system; What typically happens is that it is common to simply drop devices that are unable to compute x amount of epochs within a specific time duration [63].

The FSVRG algorithm's goal is to do one full calculation, then there are a lot of updates on each client. The updates are done by going through random permutations of data, performing a single update. The actual focus of the FSVRG algorithm is sparse data. Some features are rarely represented in the data-set [63].

The authors compared the mentioned algorithms on the MNIST data-set. Upon analyzing results the authors concluded that the FedAvg algorithm had better performance on the MNIST data-set [63].

Another algorithm that has been proposed for FL is called FedProx. FedProx and FedAvg are both similar since they both require that groups of gadgets are chosen at each iteration. Local updates are executed and grouped together to produce a global update. FedProx is meant to be a modification of the original FedAvg algorithm. FedProx makes simple modifications that allow for even better performance and better heterogeneity. The reasoning behind this is that a variety of devices that get used for FL will often have their own device constraints, so it would not be ideal or realistic to expect all the devices to perform the same amount of work. The authors that proposed the FedProx algorithm claim that the algorithm allows for different amounts of work to be performed by taking into account different devices' performances at different iterations. More specifically, the algorithm tolerates partial work instead of uniform work. By tolerating partial work, heterogeneity of the systems is accounted for, and there is more stability in comparison to the default FedAvg algorithm. To verify the effectiveness of FedProx, the authors used different kinds of tasks, models, and four data-sets. Synthetic data and simulated heterogeneous systems were also used for evaluation. Upon experimentation, the authors were able to validate their proposed algorithm's effectiveness [64].

The authors Hongyi Wang, Yuekai Sun, *et al.* proposed an algorithm for FL called FedMA (Federated Matched Averaging). FedMA was intended for federated learning of recent neural network frameworks. In FedMA, global models are constructed via layers, and by matching and averaging hidden elements with the same features. Regarding the layers in FedMA, there are some steps that are involved. Firstly, the data center gathers the weights of the first layers via clients and performs one-layer matching for obtaining the first layer weights of the federated model. The data centers then send these weights to the clients, which then train all the other layers on their data-sets. This procedure is repeated up to the last layer, a weighted average is calculated based on proportions of data for based on proportions of data points per client. Another aspect of FedMA is communication. With this aspect, clients obtain the global model at the beginning of a new round, then modify their own local models with sizes equivalent to the original models. As a result, sizes can be smaller thus easier to manage. The FedMA algorithm was implemented in PyTorch and simulations of a federated learning environment were used. A total of four data-sets were also used for experimentation as well. Upon evaluation, the authors concluded their proposed algorithm outperformed other algorithms. Furthermore, one benefit of FedMA is that it can handle communication more effectively than other algorithms [65].

Interestingly, there have been algorithms for FL that have been applied to a specific industry. Two examples come from [66] and [67]; Both of these works have proposed algorithms for the medical industry. In [66], The proposed algorithm for FL is called LoAdaBoost, shown in Fig. 12. LoAdaBoost was meant for handling medical data. After all, medical data itself is often stored on different devices and in different locations. The authors of LoAdaBoost claim that other FL algorithms only focus on modifying one issue, such as accuracy, but none of them take into consideration the computational load that various clients could be dealing with. LoAdaBoost considers three issues with FL: the client's computation complexity, the communication cost, and accuracy. LoAdaBoost is also supposed to help clients' performance too, and facilitate communication between FL's clients and servers. For evaluation, the authors compared LoAdaBoost against the baseline FedAvg algorithm. The LoAdaBoost algorithm managed to not only outperform the FedAvg algorithm but also managed to improve communication efficiency.

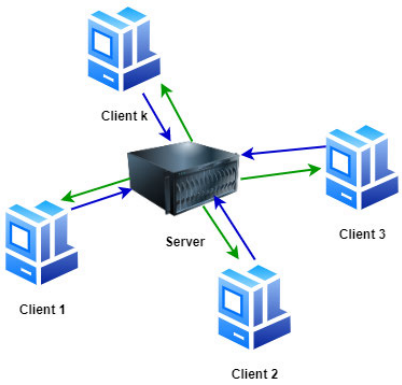

FIGURE 12. LoAdaBoost client-server architecture.

In [67], another algorithm that was applied in a medical setting comes from the authors Abhijit Guha Roy, Shayan Siddiqui, et al, where they propose an algorithm called BrainTorrent [67]. The concept of BrainTorrent is shown in Fig 13. According to the authors, BrainTorrent operates in a peer-to-peer environment. In the peer-to-peer environment, the goal of BrainTorrent is to have all centers talk with each other, instead of relying on the main server in traditional FL. BrainTorrent was meant to provide help for mobile device users. Another element that makes BrainTorrent stand out from other algorithms discussed in this section is that BrainTorrent is server-less; Typically where there can be millions if not billions of users, it is normally very convenient to

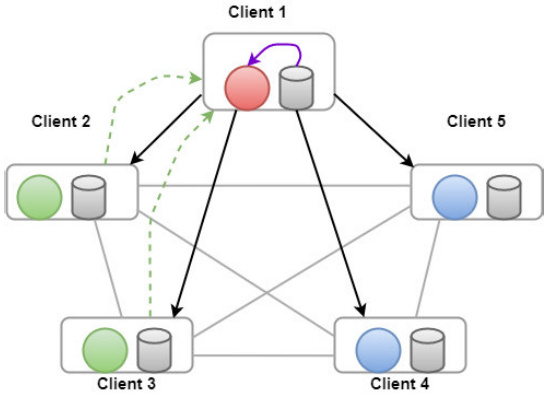

FIGURE 13. Architecture of BrainTorrent algorithm.

have the main server. The main goal of FL is to reduce costs for communicating. The motivations to do so are different for collaborative learning within communities for four main reasons:

1) The amount of health facilities forming communities is much lower.
2) Centers would be required to have stronger communication, so there is minimal latency.
3) Going off the second reason, in the context of healthcare settings, it would be difficult to have a central server because every center would likely want to communicate directly with the rest of the servers.
4) By relying on servers, if a problem were to occur, the entire structure would be rendered non-functional. That would be dangerous in a medical setting.

By not having a central server in the algorithm's framework, BrainTorrent is resistant to failure and there is no need for a central server that everyone trusts. So a client at any point in time can begin an updating process. This leads to more interaction taking place, and higher accuracy. Along with the models, each client has to maintain its own model version and latest model versions used during training. The training process at any step is conducted with the following steps:

1) Train all clients via a local data-set.
2) Have a single client at random initiate the training process. To do so, the client sends a request to other clients for obtaining the most recent models.
3) All the clients with updates send in their weights and training sample sizes.
4) A group of models is combined with the random client's recent framework via a weighted average. Then we return to the first step.

To evaluate the effectiveness of BrainTorrent, the authors used the Multi-Atlas Labelling Challenge (MALC) data-set. The data-set consisted of 30 brain scans from various people. The authors used the ADAM optimization method as well. The authors confirmed that BrainTorrent managed to outperform the traditional FL system.

V. OPTIMIZATION TECHNIQUES TO FEDERATED LEARNING MODELS

There have been some optimization techniques proposed for FL models [68] as seen in Table 4. One of the studies on this topic comes [69]. According to them, in FL mini-batch gradient descent is typically used. Mini-batch gradient descent is used to optimize models. It essentially partitions the training set into tiny batches, which get used to calculate the error of our model(s) and update the models' coefficients [70]. With mini-batch gradient descent, we are trying to find a compromise between efficiency and how comprehensive the gradient descent method should be. Using this method has several advantages and drawbacks [71], [72].

TABLE 4. Optimization techniques to federated learning models.

Technique	Advantage(s)
Mini-batch Gradient Descent	Efficiency Frequent Model Updates
Ada-grad	Faster More Reliable
RMSProp	Low Memory Requirements
Fed Adagrad	Adaptability
FedYogi	Adaptability
FedAdam	Higher Accuracy

In addition, other optimizers, such as Ada-grad, RMSProp, and Adam have also been applied. These optimization methods have been modified for Federated Learning. The RMSProp method is an algorithm that is gradient-based. A benefit of using the RMSProp Method is that it can automatically tune the learning rate, so we do not need to manually do it [73]. The Adam optimization method is a combination of both SGD and RMSProp. The method was presented by two people in their paper [74]. Since Adam combines both from two optimization methods, it has a multitude of advantages, including low memory requirements and it still works well even if there is a little tuning of parameters [73].

Additionally, the study by [75] proposes a variety of optimization methods, namely FedAdaGrad, FedYogi, and FedAdam. For the FedAdaGrad Algorithm, there are three steps: initialization, sampling subsets, compute estimates. The other algorithms, FedYogi and FedAdam have the same structure, but the difference lies in their parameters. For FedYogi and FedAdam they rely on the degree of adaptivity, meaning how well can the algorithms adapt. Having smaller values for their parameters indicate high adaptivity. Implementation of FedAdaGrad, FedYogi, and FedAdam was done in Tensor Flow Federated (TFF). The three optimization methods were compared against the standard FedAvg algorithm. The algorithms managed to have higher accuracy than the standard FedAvg algorithm.

Interestingly, regarding the optimization of FL, there is a lot of optimization for FL regarding wireless networks [76], [77]. This is because wireless networks often face some

constraints, such as bandwidth for example. Several works such as [78]–[80], and [81] have attempted to solve such issues, with mostly successful results. However, there is still an opportunity for designing guidelines and frameworks for wireless networks to accommodate FL. The work of [82] attempts to design such a framework for FL's incorporation into wireless networks for improving communication. Upon the implementation and testing of their framework, results indicated that loss can be reduced between 10 to 16 percent. This is crucial since the current FL framework does not account for wireless networks and healthcare [83]. Other works, such as as [84]–[86], and [87] have also attempted to incorporate FL into different types of wireless networks, with some of these works attempting to apply FL into IoT. In addition, [88] manages to use FL for detecting malicious clients, which will likely become important when considering the reliability of participants involved in FL to behave accordingly. Because of how much research is into FL's optimization, we can expect that FL will be able to grow in potential, applicability, and availability, thus leading FL to be more mobile [10], [89].

VI. SECURE NETWORK PROTOCOLS IN SUPPORT OF FEDERATED LEARNING

Network protocols are not entirely new. They have been around since the 1970s and 1980s, with these network protocols being modified over time to reflect current trends [90]. In the context of Federated Learning, there have been proposed network protocols and methods to tackle a few issues with FL, mainly related to privacy and traffic flow [91]. In addition, network protocols for FL have been mainly focused on wireless networks, which are essential for any industry, especially for health/medical care [92]. A summary of the network protocols is provided in Table 5.

TABLE 5. Optimization techniques to Federated Learning models.

Network Protocol	Focus	Benefit(s)
Hybrid FL	Communication Resource Scheduling Accuracy	Accuracy
FedCS	Clients' training process	Higher accuracy Robust models
PrivFL	Mobile Networks Data Privacy	Accounts for threats
VerifyNet	User privacy Integrity	Receptiveness High security
FedGRU/JAP	Traffic flow prediction	Reduced overhead

The Hybrid-FL protocol is supposed to make learning non-IID data easier. In applying FL for networking, there are some common problems FL runs into. One problem is resource scheduling, due to how different clients can be and bandwidth constraints. Because clients often have wireless linking qualities and computation limits, communication is essential so they can communicate and update any model(s) accordingly [93]. So, the FL operator has to coordinate in terms of which client(s) will take part in the FL system. While resource scheduling has been studied extensively, resource scheduling for networks in the context of FL is still new, thus a challenge. In FL, we need to look at how the client's data is distributed and consider communication capabilities, computation capabilities, and quantity of client data [94]. How Hybrid-FL works are that the server updates a model via the clients' gathered data. Then it combines that model with other models that were trained by other clients. There is a server-client architecture in the Hybrid-FL protocol, shown in Fig. 14, where the server handles the scheduling of clients, by considering the data distribution and channel condition of each client. The Hybrid Protocol's main idea is while a few clients transmit their information to the server, both the clients and server will update models. The protocol has a Resource Request step, where clients are asked about how much data they have, their computational and communication properties, and if they allow their information to be transmitted to the server. That way the time required for updates can be estimated. There is also a distributing step, where a model is transferred for choosing a few clients so that local updates can take place. Upon testing the proposed protocol's effectiveness, the authors noted that Hybrid-FL was able to increase overall accuracy in regards to data distribution [94].

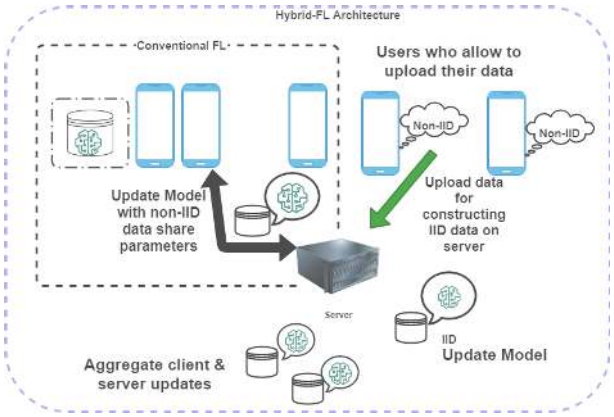

FIGURE 14. Hybrid-FL framework.

Like the Hybrid-FL protocol, FedCS was intended to tackle problems associated with FL. Specifically, the problems FedCS attempts to solve deals with the clients' training process. While the original FL protocol's clients are free from revealing their own data, sometimes these clients can have limited computational resources, so the training process for these clients can be inefficient [95]. By attempting to remedy this, we can cope with many updates while considering the clients' restraints. The authors note that FedCS was built using a MEC framework [96]. Regarding FedCS's framework as shown in Fig. 15, there are a few key steps:

- **Client Selection** - In the original FL protocol, clients are randomly selected. In the FedCS protocol, there are two additional steps. First, there is a *resource request* that asks random clients to indicate their statuses on different

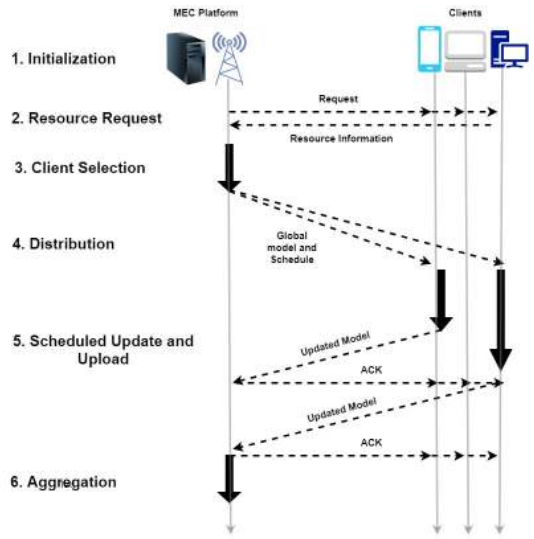

FIGURE 15. FedCS protocol framework.

things, such as channel states. Then, the client selection receives this information, which is used to estimate how much time is needed for the next two steps. The clients' information is also used to determine which clients can proceed to the next two steps.

- **Distribution** - In this step, a global model gets transmitted.
- **Update and Uploading Schedules** - Here, the model is updated via clients. The clients transmit the new parameters to the server. A server gathers up clients' updates and measures the models' performance.

To verify FedCS's effectiveness, the authors used two data-sets: the CIFAR data-set and the Fashion MNIST data-set, both of which were focused on object classification. FedCS was compared against the default FL protocol. From analyzing the results, FedCS was able to obtain higher classification accuracy and was able to generate high performing models in shorter amounts of time [96].

In addition, a protocol called PrivFL was introduced, focusing on privacy-preservation and is geared toward mobile networks. With PrivFL, what is being considered in regard to privacy is the mobile users' data privacy from the server and the model privacy against the users. Interestingly, the authors modified their proposed protocol for three regression methods: Linear, Ridge, and Logistic Regression. With the PrivFL protocol, there is a security aspect to it, which the authors demonstrate by using three different threat models. PrivFL was also evaluated by using eleven data-sets [97]. Regarding PrivFL's structure, there are two key elements: a server, and groups of mobile users that are connected to a mobile network. The server is responsible for training. The training process has a few factors:

1) **Correctness** -For ensuring accurate inputs from users, PrivFL needs to output the right model during training. If the server manipulates the model, or if the users use incorrect inputs, then the model lacks correctness.
2) **Privacy** - PrivFL's objective is to ensure the users' privacy is not compromised when they input something and to protect the server's privacy. So the server should not memorize any information about the users' inputs. Similarly, users should not learn any information regarding the model.
3) **Efficiency** - The server performs most work in the training phase. Therefore, the computation and communication costs need to be minimal.

Overall, the PrivFL protocol is both comprehensive in its implementation and structure. The protocol also considers different threat models, which make it a reliable protocol overall [97].

Furthermore, another protocol for FL comes from the authors Guowen Xu, Hongwei Li, *et al.* Their proposed protocol is called VerifyNet, which is shown in Fig. 16. VerifyNet tries to protect the users' privacy for the process of training and verify the reliability of the results that are sent back. The issue that the authors' highlight is that while there are several approaches focusing on security and privacy, there is still a problem of clients being able to verify if a server is functioning properly without compromising the users' privacy [98]. As such, the overall goal of this protocol is to address three major problems that federated training often runs into:

1) How to protect the users' privacy in workflow
2) Spoofing
3) How to offer offline support for users

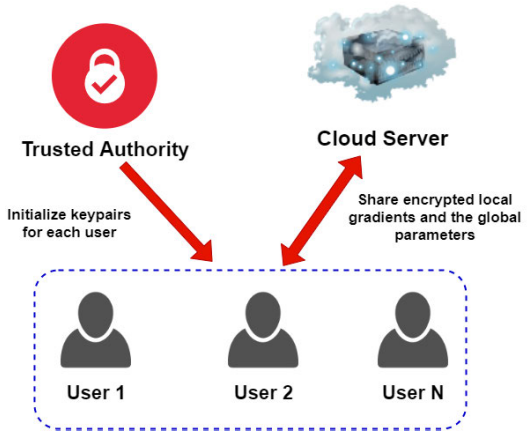

FIGURE 16. VerifyNet architecture.

There are five rounds in the VerifyNet protocol that are: Initialization, Key Sharing, Masked Input, Unmasking, and Verification, with Fig. 16 summarizing VerifyNet's architecture. First, the system is initialized, then, public and private keys get generated. Each user then needs to encrypt their local gradient and send it to the server(s). When the server receives messages from all the users online, it gathers all of the online users' returns results. Users can decide to accept or reject the results. The authors tested VerifyNet by utilizing 600 mobile devices in order to evaluate the protocol's performance. The devices have 4GB of RAM

and had Android 6.0 systems. The data used for evaluation came from the MNIST database which had sixty-thousand examples for training and ten-thousand examples for testing. The authors noted that VerifyNet was receptive to users dropping out of the FL learning process. VerifyNet was also able to achieve high security, but unfortunately, VerifyNet has high communication overheadm [98].

Another protocol comes from the authors Yi Liu, James J.Q. Yu, *et al.* Here, they propose a protocol for FL that is geared toward traffic flow prediction [99]. The protocol aims to achieve an accurate prediction of traffic while still being able to preserve privacy. The algorithm is called the Federated Learning-based Gated Recurrent Unit neural network (FedGRU). The actual protocol designed is called the Joint Announcement Protocol. Traffic flow prediction provides information about traffic flow, such as traffic history to predict future traffic flow. Privacy concerns exist in predicting traffic flow too as seen in Fig. 17. Regarding the FedGRU algorithm, FedGRU aims to provide traffic flow prediction. There are about four steps to it, which are the following:

1) Initiate the model by pre-training it. This relies on public data-sets that we can use without privacy concerns.
2) A copy of a global model is sent to institutions, and they train their copy on local data.
3) Then, each of the institutions sends model updates to the Cloud. No private data is shared, just the parameters which are encrypted.
4) The cloud gathers all the updated parameters uploaded by all institutions to generate a new model, and then the new global model is distributed to all participating institutions.

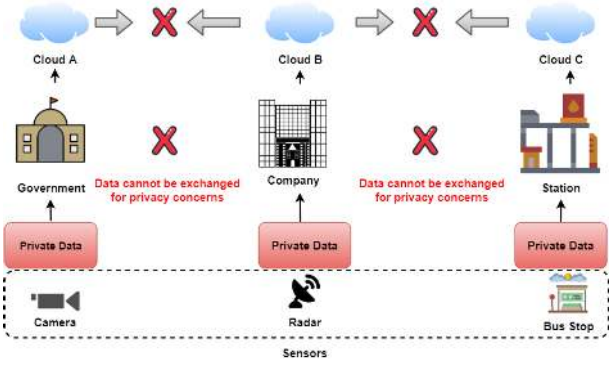

FIGURE 17. Traffic flow diagram. This illustrates how difficult traffic flow prediction can be due to data not being able to be shared.

Then, regarding the Joint Announcement Protocol, we need to keep in mind the number of participants in FedGRU because there is a small number of participants. [99] Since there is a small number of participants, we can think of the Joint Announcement Protocol as a small-scale FL model. The participants in the Joint-Announcement protocol are the institutions and the cloud. There are three phases in the protocol:

1) **Preparation** - When given an objective to complete, the institutions that are voluntarily participating will check-in with the Cloud.
2) **Training** - The cloud sends in the pre-trained models first. Afterward, it sends a checkpoint to the institution. The cloud needs to randomly select a fixed ratio of institutions to partake in training. Each institution trains data locally and sends parameters.
3) **Gathering** - In this phase, the cloud gathers up all the parameters sent in to modify the model. Then the cloud modifies the global model by sending in check-points. After this, the global model transfers all the updated parameters to each institution.

To evaluate their proposed methods, the authors used data from the Caltrans Performance Measurement System PeMS) database. Traffic flow information was obtained from over 30,000 detectors. The authors chose a few metrics, mainly the Mean Absolute Error (MAE), Mean Square Error (MSE), Root Mean Square Error (RMSE), and Mean Absolute Percentage Error (MAPE) for testing predicting accuracy. Results obtained proved that their proposed methods were able to perform comparably to the other competing methods. In addition, the Joint Announcement Protocol was able to reduce communication overhead by sixty-four percent [99].

VII. CHALLENGES AND LIMITATIONS OF FEDERATED LEARNING

Despite FL having many benefits, FL is still prone to many limitations and there are still challenges that hinder FL from being fully adopted by various industries. These challenges mainly are regarded in terms of privacy, security, and in some cases the technical requirements [100], [101] [102]. Table 6 summarizes various challenges of FL.

When using data for training in Federated Learning, the data itself is often imperfect. The data could be imbalanced, and some classes in data-sets could be missing entirely [103]. With these drawbacks, the end result is models that are highly inaccurate. Typically, training data is skewed quite heavily [104]. The training data's characteristics can vary by a lot due to being spread at different sites. In these scenarios, it would not be beneficial to have a naive implementation of an FL model because the resulting model would be inaccurate [105]. To remedy the data skewing problem, we would need to rely on the server [106]. Furthermore, data skewing can even affect different kinds of data that can be used in FL. One example is tabular data [107]. Overall, data skewing is due to four primary factors: data imbalance, missing classes, missing features, and missing values.

- **Data Imbalance** - This issue happens when one or both entities have more (or less) training data than another entity. For instance, one partner has thousands of training samples for class A and the other partner has a few tens of training samples for class A. When this happens, the resulting model built via FL can be very poor So techniques are needed in order to remedy data imbalance issues.

TABLE 6. Summary of existing challenges for FL.

Reference No.	Challenge(s) of FL
[117] - [121]	Data skewing Data Imbalance
[122] - [126]	Communication Systems Heterogeneity Privacy
[127]	Hardware Power Network Connectivity
[128] - [133]	Security Privacy
[134] - [139]	Model Performance Accuracy Security Privacy
[140] - [149]	Privacy Security Performance Trust Accountability System Architecture Trace-ability

- **Missing Classes** - This can happen when one or both entities have training data that represents a class, but that class is not present in the other entity's training data.
- **Missing Features** - Missing features occur when one or more entities have training data containing features that do not exist in the other entity's data. If we were to create a model, the resulting model would not be able to know the missing features, or the model could be incomplete because it requires features that are missing.
- **Missing Values** - With this issue, this particular issue happens when one or both entities have training data where some values are missing.

In addition to the data skew problem previously mentioned, there are other challenges that FL is prone to, such as Communication, Systems Heterogeneity, and Privacy. Communication is essential for FL, especially in the healthcare and medical care settings [108], [109]. Combined with privacy concerns over raw data being sent, there is the demand that the devices need to remain local [110]. FL involves a vast quantity of devices, so communication can be slower by a lot. In order to formulate a model, we need to design methods that are communication-efficient [111]. This means that we send over smaller messages or smaller model updates for the training process, instead of just sending the entire training data-set [112]. For this, we need to consider two factors: reducing the number of rounds for communication and reducing the size of each message for each round.

Additionally, because FL can involve a lot of different devices, each device is different in terms of hardware, network connectivity, and power [113]. Depending on the devices' configuration, network size, and system constraints usually result in a small portion of the devices operating. The devices themselves could be untrustworthy, and it is not unusual for a device to suddenly leave at a certain point because of connection or energy limitations [112]. As such, FL methods that are developed need to take into account a few factors:

1) How to anticipate there are low amounts of participation
2) Tolerating different device hardware
3) Robust methods we need to account for these dropped devices in network settings.

In FL, privacy is often the biggest concern. In the training process, communicating model updates can reveal sensitive information [114], [115]. Even though there have been newer approaches proposed for tackling privacy, these methods can reduce model performance and accuracy. Being able to find a compromise with this trade-off is still a challenge when attempting to solve the privacy issue [112], [116]. The authors of [117] also note that the biggest issues in FL are mainly security and privacy. As such, efficient FL algorithms that deliver models with high performance and privacy protection without adding computational burden are desirable [118], [119]. Because local models are trained via newer data to highlight new updates, it is likely that adversaries can influence the local training data-sets to compromise the models' results. There also several attacks that FL can be prone to. So being able to formulate algorithms and methods that can protect themselves from these attacks are crucial so that model performance and accuracy do not suffer [120]. Additionally, the authors of [121] also note that security and privacy are the biggest challenges in FL. Regarding privacy, it is possible to extract information from a trained model. When training, the connections that are noted in the training samples are aggregated inside the trained model. So, if the trained model is released, what happens is unexpected information reveal to hackers. It is also possible to obtain data of a victim by instantiating requests to the model [122]. This occurs when someone gains unauthorized permission to make prediction requests on a model that has been trained. Then, the adversary can use the prediction requests to extract the trained model from the owner of the data.

In terms of security, FL is prone to a variety of attacks that can compromise the model performance and accuracy [123], [124]. One attack is called the Data-Poisoning attack, where a person can tamper with the model by creating poor-quality data for training that model. The goal here is to generate false parameters. These types of attacks can achieve high misclassification, up to ninety percent. There can be different modifications to the data-poisoning attack too [125]. One example is a Sybil-based Data-Poisoning attack, where the adversary boosts the effectiveness of data poisoning by creating multiple adversaries. Another type of attack is Model-Poisoning, which is more effective than Data-Poisoning. In Model-Poisoning attacks, it is possible for the

adversary to modify the updated model. Another type of attack that FL can be prone to be called a Free-Riding Attack, where the adversary wants to leech benefits from the model without being part of the learning process. This results in legitimate entities contributing more resources to the training process [121].

In the context of other settings, the challenges, and limitations of FL are somewhat similar, and there may actually be more challenges and limitations to applying FL for these other settings [126]. Several challenges and limitations include Privacy and Security, Security, Performance, Level of Trust, Data Heterogeneity, Trace-ability and Accountability, and System Architecture. We often work with sensitive data that needs to be guarded accordingly. While one of FL's purposes is to protect privacy, FL is not able to solve all privacy issues [127]. Because of strict regulations and government policies regarding medical data, leakages of private information or the possibility of information leakage are unacceptable. Even more challenging is that there are a lot of different regulations on medical data, so there is no one-size-fits-all solution [126]. In terms of trust, this mainly deals with the possibility of information leakage and preventing that from occurring [128]. Trust is needed in the medical and healthcare industries so that confidence in FL and FL's performance can be established [122]. Regarding trust, there are two types of collaboration participating FL entities can be in:

- **Trusted** - This is mainly for FL entities that are considered reliable and are bound by a collaboration agreement. So we can eliminate a lot of malevolent attempts. As a result, the need for counter-measures are reduced, and we can rely on the typical collaborative research.
- **Non-Trusted** - For FL systems that operate on larger scales, it can be impractical to establish an enforceable collaborative agreement that can guarantee that all of the parties are behaving accordingly. Some entities could try to degrade performance, bring the system down, or extract information from other entities. In this particular setting, security strategies are needed to mitigate these risks such as encryption, secure authentication, trace-ability, privacy, verification, integrity, model confidentiality, and protection against adversarial attacks.

With Data heterogeneity, data is very diverse in terms of type, dimensionality, and characteristics. We also have to consider the task(s) due to demographics. [129], [130]. This can pose a challenge within the FL frameworks and methods. One of the biggest assumptions in FL is that the data is IID (Independently and Identically Distributed) across entities [131]. Another concern is that data heterogeneity may lead to a scenario where the global solution may not be the optimal final solution [132].

Furthermore, the reproducibility of a system is essential in industries such as healthcare. A trace-ability requirement should be fulfilled to make sure that system events, data access history, and configuration changes can be traced during the training processes [27]. Trace-ability can also be used to note the training history of a model and avoid the training and testing data-set overlapping [133], [134]. In cases where there is non-trusted collaboration, traceability, and accountability processes both need integrity. After the training process reaches the agreed optimal model guidelines, it may also be helpful to measure how much each entity contributed, such as resources consumed. We also need to consider the system architecture. Various institutional entities are usually equipped with better computational resources and reliable and higher throughput networks [28]. As a result, this allows us to train larger models with larger amounts of training steps [135]. This way we can share more model information. FL is not perfect, but with more research and implementation, these challenges will hopefully be minimal. Overall, there are many challenges and limitations of FL that still need to be considered such as Data Skewing, Communication, System Heterogeneity, Privacy, Security, Performance, Level of Trust, Data Heterogeneity, Accountability, and System Architectures.

VIII. MARKET IMPLICATIONS OF FEDERATED LEARNING

While not explicitly researched, FL is anticipated to have considerable implications for the marketplace, especially in industries that in critical need of newer technology such as the healthcare and medical markets [136]. According to CMS.gov, national healthcare spending is expected to increase by an average rate of 5.7% per year. At that rate by 2027, healthcare spending will reach nearly six trillion dollars. In addition, CMS also noted that hospital spending is forecasted to have increased by 4.4% back in 2018, but this number is higher in 2020, and will likely continue to increase over time. These statistics indicate that the medical and healthcare industries heavily impact the market by a lot [137].

It has also been noted that the healthcare industry produces more data than a few industries, such as advertising. The healthcare industry is producing about 30% of the world's data if not more. As such, healthcare is expected to keep flourishing through 2025 [138]. So in terms of the market impact, healthcare and medical data are expected to grow quicker than the manufacturing, financial services, or media industries. Interestingly, the IDC(International Data Corporation) used an index called DATCON(DATa readiness CONdition) to assess the data management, utilization, and monetization of various industries. There are a few metrics used such as data growth, security, management, and involvement. Scores of DATCON range from 1-meaning that the industry lacks the capabilities to manage and monetize its data, and 5-meaning that the industry is completely optimized [139]. The healthcare industry has a score of 2.4, meaning that they fall below average in a few data competency metrics such as technological innovation. Because technology innovation is low in both healthcare and medical industries, the industries themselves fail to tackle recent data management problems, and are unable to invest in more advanced architectures [140].

Given these statistics, FL could greatly impact the market of healthcare. FL could also impact the medical industry as well. By 2025, the market size for healthcare and medical industries will reach approximately seventy billion U.S. dollars. With a market size that huge, FL will likely generate a positive impact [141]. These industries both rely on data management and privacy regulations in order to keep their operations functioning properly [142]. With FL introducing new ways to manage data without risking privacy, the market implications are likely to be monumental, leading most if not all industries to generate more revenue, which could be potentially used to obtain more advanced frameworks to better handle data. Several companies are also invested in data as well [143]. These companies include but are not limited to:

- IBM
- Microsoft
- Oracle
- SAP
- SAS Institute

IX. BENEFITS AND COSTS OF FEDERATED LEARNING

Because various industries are directly impacted by regulations regarding privacy, FL is a promising technology to alleviate these privacy concerns [144]. One of the biggest reasons that FL is gaining a lot of popularity is because most data stays on the consumer's device. Furthermore, FL offers a new method of the framework, since traditionally, we would rely on a centralized architecture for training algorithms [145]. FL offers a decentralized solution, which not only is more efficient, it can also lead to higher performance models and privacy is not an obstacle to training algorithms with large data-sets. However, in order for us to fully apply FL in all industries, we need to weigh the benefits and costs carefully [146].

Regarding FL's benefits, there are a lot. One of the biggest benefits FL has to offer is its mitigation of privacy concerns [147]. FL is able to account for privacy concerns such as data access and data management. As a result, it is most optimal for industries where privacy is a huge issue. Also due to the decentralized approach FL has, we do not need to worry about actively training the algorithms ourselves very much. The algorithms we use for FL train themselves directly on the devices and only transfer back the relevant data that is needed [148]. As a result, the need to use the user's data is remedied and the training process can be more streamlined. Another advantage of FL is its flexibility. The learning process for FL can be conducted when the devices are charging, connected to WiFi, and/or not in use. So users will not need to worry about wasting their data or battery [149]. Overall, the major benefits of FL can be summarized in the following points:

- **Data Security and Privacy** - FL training happens on the devices, and only the models get transferred. Due to this, it solves the issue of storing huge quantities of sensitive or personal data at a central location.
- **Real Time Prediction** - Because the device itself is where prediction happens, we do not have to worry about any time-lag that happens when sending input back to a server and then sending results back.
- **Offline Prediction** - Prediction can work even when the devices are offline. So there is no stressful need to worry about the device having an internet connection. As long as a device can get input, the predictive models can be utilized to do their work.
- **Minimal Infrastructure** - FL does not require much intensive hardware, in fact, the requirement is minimal.

Unfortunately, FL still has some major setbacks that need to be fixed so that FL can be applicable to all industries [150]. One cost of FL is that it can require more device power and memory to train the models, which could pose problems for devices that have lower power and memory [151]. Another cost is bandwidth. Typically in machine learning, we use data centers, but for FL, we would need to rely on WiFi. WiFi's bandwidth rates are lower, and bandwidth to devices has not grown as quickly as their computation power [152]. Having insufficient bandwidth for FL can result in inefficient communication, latency, and slower learning processes. Another challenge of FL regards the reliability of devices to participate in the FL learning process. The learning process of FL is iterative, so it relies on the devices involved to constantly communicate over iterations until the learning process stops [153]. However, in real-world scenarios, not all devices may fully take part in the complete iterative process from start to finish. If a device were to drop out in the middle of the learning process, their data cannot be used. As a result, the learning quality of FL could be compromised. A few other costs of FL are as follows:

- **Non-independent identically distributed data (Non-IID)**
- **Unbalanced data-sets**
- **Large-scale distribution**

Overall, FL has a lot of promising potentials that can be utilized by various companies for various reasons. FL is not perfect, unfortunately, so we need to carefully consider what devices to use and how we set up our infrastructure to get the most benefit of FL.

X. APPLICATIONS AND USE-CASES OF FL

Federated Learning is considered one of the most exciting technologies to date. Throughout this paper, we have spoken about the definition of FL, the current platforms and architectures surrounding FL, and discussed its benefits and costs. However, we have not talked about the applications and use cases of FL. Many industries and companies are beginning to incorporate FL into their own work cycles and products. As a result, many applications of FL have surfaced. Some of these applications and use-cases will be discussed in this section. These are applied to various purposes, including healthcare.

A. APPLICATIONS

1) GOOGLE KEYBOARD QUERY SUGGESTIONS

The authors of [154] managed to apply FL in an attempt to improve the quality of keyboard search suggestions. This was for the Gboard, which is a virtual keyboard for mobile devices. Gboard comes with plenty of features, such as auto-correction, next-word prediction, word completions, etc. The authors note that since Gboard is both a mobile application and a keyboard, Gboard has unique constraints that make FL suitable for it. Gboard needs to not only respect the consumers' privacy but also not have any latency, which is crucial for a mobile setting. The authors first collected training by observing how consumers interact with Gboard. Because this particular application of FL is mobile-based, the authors had to ensure the consumers data usage and user experience were not negatively impacted. So they had to rely on Android's Job Scheduler for managing background operations when the devices are idle or charging. The authors also managed to build a client-server architecture. The server waits until a certain amount of clients have connected, then provides each client with a training task. The clients are in charge of executing those tasks. To manage the load across the devices, the client is notified of how long it should wait before communicating with the server again. The authors were able to fully build and train their model successfully for improving keyboard search suggestions.

2) MOBILE KEYBOARD PREDICTION

Like the previous application, FL was also applied for keyboard prediction and keyword spotting [155]. However, the approaches are different. Here, the authors managed to train a Recurrent Neural Network (RNN). Their RNN was deployed on a framework FL. Mobile keyboard models are constrained in multiple ways. The models need to be small and be able to run on low and high-end devices. Consumers usually expect a visible keyboard response within about 20 milliseconds of an input event. Due to how frequently mobile keyboard apps are used, client device batteries could be rapidly depleted. The authors managed to use what is called the Coupled Input and Forget Gate (CIFG) network, which is a type from the notable LSTM (Long Short Term Memory) RNN. CIFG uses a single gate for controlling both the input and recurrent cell self-connections, thus shrinking the number of parameters by twenty-five percent. By using CIFG, the authors note that CIFG is beneficial for mobile devices due to the reduced quantity of computations and parameters. The authors used TensorFlow for training the model. In addition, FL was used for training the server-side of CIFG. This particular training relied on data from Gboard users who have opted to share bits of text while typing in Google apps. The text is modified to contain short phrases of a few words. For this application, 7.5 billion sentences are used for training, while the test and evaluation sets each have 25,000 sentences. The average sentence length in the dataset is about four words. For the FL training, the authors stored data in local caches on devices. However, in order for the clients to partake in FL, they needed to meet a certain amount of requirements. The devices need to have at least 2GB (gigabytes) of memory available. Also, the clients are only allowed to participate if they are charging, connected to a network, and idle. Upon testing, the authors were able to have their CIFG network trained on FL achieve optimal performance [156].

3) RANKING BROWSER HISTORY SUGGESTIONS

Here, FL was applied towards ranking browser history suggestions, specifically for the Firefox browser. The authors used FL to train a model based on user-interactions while focusing on privacy. When implementing their model, the authors also focused on robustness, so for optimization, they decided to use a variant of the RMSProp optimization technique. The reason that FL can be difficult to deploy in cases like this is that there is a higher cost of testing out different versions of whatever FL system is built. Each experiment with real users consumes more time and can lead to negative user experience. When applying FL for ranking browser history suggestions, the authors deployed their system to a large number of Firefox users. In less than a week, those users were able to help train and evaluate the new model for Firefox's improved URL bar. The new model leads to users typing over half a characterless. The authors managed to prototype their work by first using simulations. That way, the authors could simulate am FL optimization process. For their simulation, the authors used a mock data-set that was designed to resemble data they expected users to generate. The authors also used client-server components for their system. The authors noted that the FL protocol they used was too simple, so improvements would be needed to make better use of data. Overall, the authors' application of using FL to improve ranking browser history suggestions was optimal and managed to preserve the consumers' privacy [157].

4) VISUAL OBJECT DETECTION

In [158], FL was applied towards visual object detection, which is a part of computer vision. Visual object detection is applicable in many ways, such as fire-hazard monitoring. Unfortunately, it is often difficult to form object detection models due to privacy concerns and high costs of transmitting video data. Notably, while FL is a promising approach to solve this issue, not everyone is familiar with FL. As a result, there is no easy-to-use tool that can enable computer vision developers who are not experts n FL to fully apply FL to their systems to gain the most out federated learning. So the authors designed a platform called FedVision in order to support FL-powered computer vision applications. Typically in object detection, the users use the data obtained from cameras and uploads to a center for training. Once a model has been trained, it can be used for various tasks. However, the disadvantage here is that the users do not have control over how the data would be used. In regards to FedVision's framework, FedVision uses a visual object detection framework based on

YOLOv3. FedVision also allows for the training of object detection models with data-sets stored locally via multiple clients. The user-interaction is designed so that there is no requirement for users to be familiar with FL. For FedVision to assist in federated model training, there are six components:

- **Configuration** - This enables users to set up training information, such as the number of rounds, quantity of reconnections, plus server URL for uploading the models' parameters and other key elements.
- **Task Scheduler** - Handles scheduling, which is used for coordinating communication between the FL servers and clients. This way resources can be balanced.
- **Task Manager** - When multiple model algorithms are being trained by the clients, this component coordinates multiple federated model training processes.
- **Explorer** - Monitors resource utilization on the clients' side. These resources can include CPU usage, memory usage, etc. The Explorer communicates to the Task Scheduler for load balancing decisions.
- **FL SERVER** - Server for FL. The Server is in charge of uploading model parameters, gathering models, and dispatching them.
- **FL CLIENT** - Client for FL. The Client hosts both the Task Manager and Explorer components and performs local model training.

There have been positive impacts on using FedVision. FedVision has been used by three major companies to develop computer-vision based safety hazard warning applications. FedVision has also managed to help consumers improve their operational efficiency, achieve data privacy protection, and reduced cost. Such an application of FL is a notable example of how FL can be fully applicable and garner a positive impact. FedVision has also managed to inspire other applications of FL in similar situations, with one example being [159], where the authors attempt to use FL for Human Activity Recognition.

5) PATIENT CLUSTERING TO PREDICT MORTALITY AND HOSPITAL STAY TIME

Unlike the previous applications of FL discussed so far, this application of FL is in a medical setting. In this case, the focus was on EMRs (Electronic Medical Records). EMRs are the most important component in a healthcare setting, being used for predicting disease rates, how a patient responds to treatment, plus other events. EMRs have been handled using traditional machine learning mechanisms, and while these traditional machine learning mechanisms have been successfully applied towards EMRs, there is a false assumption that EMRs can be easily stored and shared in centralized locations. This assumption is false because EMRs are generated by patients in various healthcare facilities and clinics. They are sensitive in nature, so traditional machine learning mechanisms are not applicable. There are concerns regarding the storage of EMRs, security, privacy, cost, and availability of sharing medical data. The authors note that while FL can greatly tackle these issues, FL may not perform optimally depending on how the data is set up. So the authors proposed an algorithm called CBFL (Community Based Federated Learning) to remedy this issue. The authors used the CBFL algorithm to predict patient mortality and stay time at the healthcare facility. CBFL was developed based on the eICU collaborative research database, which has data of a little over two-hundred thousand patients admitted to two-hundred hospitals. There are three procedures in the CBFL algorithm, which are: encoder training, K-Means Clustering, and Community-Based Learning. Results indicated that CBFL achieved high accuracy and out-performed the standard FL model. [160]

6) DRUG DISCOVERY

Surprisingly, FL has also been used for drug discovery. At least two works, one by [161] and another by [162]. Both works cover FL's applicability for drug discovery, however, these two works tackle specific problems related to drug discovery and apply FL to solve them. In [161], the focus on incorporating FL for drug discovery deals with data-sets with high biases. Biased data can be a huge problem for model training because the data itself can be skewed due to these biases. So this results in a model that may be inaccurate. The authors managed to form a general FL framework for drug discovery. The general FL framework consists of server-coordinator and collaborators for the FL client. During each round of training, the framework's server transmits the newest model to each of the clients. They handle model updates by executing training. The clients also encrypt and transfer the model updates via a protocol. Then the coordinator-server gathers model changes, then uses them to update the model. The authors managed to evaluate their proposed framework by comparing it against centralized learning and used seven drug-related data-sets. Their framework managed to outperform centralized learning.

In [162], the focus is on using what is called the Quantitative structure-activity relationship (QSAR) and applying FL to it for drug discovery. QSAR analysis is typically used in, and having various facilities collaborate usually leads to better results. Unfortunately, there is still hindrance regarding intellectual property, which can hinder collaboration for drug discovery via QSAR. QSAR is typically used to investigate and predict various properties of compounds, which is crucial for drug discovery. Shaoqi Chen and his team proposed a new platform for FL-based drug discovery called FL-QSAR. FL-QSAR uses horizontal FL architecture. For testing, Shaoqi Chen and his team used fifteen data-sets. They also made comparisons of results from incorporating horizontal FL versus not incorporating horizontal FL, collaborations via horizontal FL and single clients, and a classic privacy-preservation framework. Results indicated that FL-QSAR outperforms single clients, and by using the horizontal FL architecture for FL-QSAR, FL-QSAR is efficient for collaboration between pharmaceutical institutions.

7) fMRI ANALYSIS

In the work of Xiaoxiao Li, Yufeng Gu, *et al.*, the focus is quality data in healthcare settings. Typically in healthcare settings, the data itself suffers from lack of accuracy and generalizability. Because high-quality data is not often available in healthcare settings due to concerns regarding reproducibility, models also suffer in performance. Patients are also concerned about their medical data being used for future health insurance decisions and shared with their employers. Health providers also worry that if their health statistics are made publicly available, they will lose patients or suffer huge consequences if they cannot assess their performance. The authors explore this dilemma by applying FL for functional MRI (fMRI) data. the fMRI data is related to different kinds of neurological diseases or disorders. The authors managed to form their proposed framework without data-sharing. The proposed framework consists of two main elements: local updates, and communication to a global server. For testing, the authors used resting-state fMRIs from the ABIDE dataset (Autism Brain Imaging Data Exchange). Here, they used it to identify autism spectrum disorders (ASDs) and a healthy control group. The proposed framework demonstrated the advantages of using FL and has possible use-cases for identifying rarer diseases with fewer patients [163].

8) BRAIN TUMOR SEGMENTATION

Another example of FL being applied for various medical settings, FL was used for brain tumor segmentation. This particular application of FL also covers medical imaging. While methods such as Deep Neural Networks have illustrated notable findings, they are very reliant on quantity and diversity of the training data [164], [165]. It is problematic because the needed training data may not be available due to having low incident rates of few diseases/disorders and a low number of people. For this application of FL, the authors of [166] used the BraTS 2018 dataset, which contained MRI scans of almost 300 people with brain tumors. The authors compared their methods against data-centralized training. The results indicated the optimal performance of the authors' proposed method.

9) DISTRIBUTED MEDICAL DATABASES: META-ANALYSIS OF LARGE-SCALE SUBCORTICAL BRAIN DATA

Here, the authors of [167] manage to propose an FL framework for accessing and analyzing biomedical data without sharing information. Specifically, the focus is on analyzing brain structure relationships across various diseases. There is a lot of data containing brain images, so there is a lot of opportunities to fully comprehend the genetic mechanisms for various brain-related diseases. Unfortunately, datasets, which are stored in unique places, cannot always be shared because of privacy and legality concerns, so we are limited in fully exploiting data for studying brain disorders. The authors' proposed FL framework was first evaluated using artificial data, then applied at multiple databases. The authors were able to validate their proposed framework was sufficient for the intended purpose. Another example by [168] was able to implement FL for a service-based authentication system specifically for imaging systems. Here, the authors were able to propose their own system for medical imaging systems in order to account for better reliability and security.

10) FedNER

In this particular application of FL, the focus is on applying FL for Medical Named Entity Recognition (NER). NER has many applications in a healthcare setting, but sufficient data that is labeled is crucial for training and obtaining an accurate NER model. Unfortunately, labeled data in the medical community is limited due to sensitivity. NER is focused on identifying various medical entities such as drug names, reactions, and symptoms from unstructured medical texts and classify them into different categories. The authors in this application propose an FL framework called FedNER to better make use of the medical data and obtain an accurate NER model without exchanging sensitive medical data. In the FedNER framework, the server communicates with multiple clients model updating and model sharing. The clients are different medical platforms. Furthermore, the authors tested FedNER by using three medical NER data-sets, and used an 80-20 ratio(80% Training, 20% Testing). FedNER was compared against a few baseline NER methods. The authors found that their FedNER model managed to outperform other NER methods, thus achieving better performance overall [169].

Overall, FL has many applications in various industries, with one example being an FL method called FedRec, where this focuses on news recommendations [170]. Other works such as as [171], [172] [173] involve the application of FL in terms of tackling FL's weaknesses, such as cooperation and data quality.

B. USE-CASES

Due to the number of applications that incorporate FL, we can build a few use-cases of how FL would be applied towards certain problems [174]. These use-cases can also be applied towards the medical and healthcare settings since these industries greatly need newer, efficient, and accurate approaches for handling their data.

1) MEDICAL IMAGING

One use-case of FL in a medical setting comes from the authors Ignacio Blanquer, Angel Alberich-Bayarri, *et al.* In their work, they attempt to develop applications on a platform known as ATMOSPHERE [175]. ATMOSPHERE is a platform where users are able to develop applications. In this case, they were gearing this toward medical imaging. What makes this unique is that this use case also deals with federated cloud infrastructures. These are usually used for handling requests to balance workloads. As such, this allows for multiple users to handle resource demand peaks that cannot otherwise be handled locally. The use cases the authors created was for early diagnosis of Rheumatic Heart

Disease (RHD). RHD is the main cause of heart halve disease, producing severe damage to the heart organ if untreated. Early detection is limited, so FL is a potential candidate to solve this issue.

Regarding the authors' model, which is shown in Fig. 18, their application is an image-analysis method that handles feature extraction, which is fed into a classifier that uses deep learning mechanisms. The model is meant to differentiate between someone definitely having RHD and someone being borderline for RHD. For training, the authors used a large database that includes over four-thousand studies. These studies contain echo-cardio videos and demographic data, and three categories are included: Definite RHD, Borderline, and Normal. For the ATMOSPHERE platform, an application contains three main components:

1) **A Virtual Infrastructure Blueprint** - Here, this involves any third-party components for handling resources and back-end execution.
2) **Application Dependencies** - These are handled via docker files. The system uses these for building images.
3) **Final Application** - This is made up of the interface layer and the processing backend.

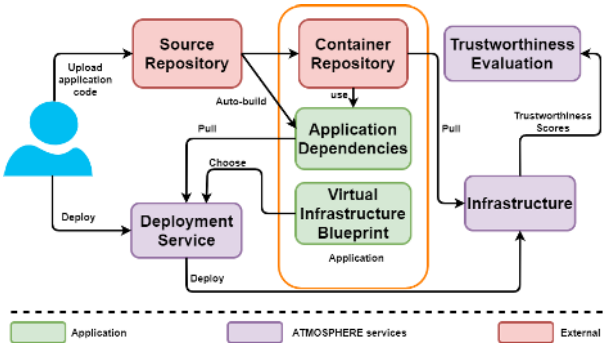

FIGURE 18. Complete framework of ATMOSPHERE.

By using the ATMOSPHERE platform for medical imaging, specifically for early detection of RHD, there are several benefits. One benefit is that there are simplified settings for training and building models. Because of this, those working on the ATMOSPHERE platform are guaranteed flexibility and a smooth transition from building models to production. There is also no need to worry about any geographical boundaries and there is still secure access for data [175].

2) ANOMALY DETECTION

Another use-case for FL comes from [176], where the authors successfully attempted to build an anomaly detection system(called DIoT) using FL to detect various IoT devices. Their system is autonomous, and can effectively operate without human intervention or labeled data. Regarding the architecture of DIoT, it consists of two components: a security gateway, and an IoT security service. The security gateway acts as a local access gateway to the Internet. It is responsible for identifying the device's type when that device gets added into the network. The security gateway component is also responsible for performing anomaly detection for identifying any compromised devices in the network. Meanwhile, the IoT security service supports the security gateway. It maintains a repository of specific anomaly detection models depending on device type. The assumptions made are that there is no malicious manufacturers, the security gateway is not compromised, and the automated identification of IoT devices. Most notably, their system was able to achieve a 95% detection rate with no false positives. Additionally, their system is able to cope with both emerging new and unknown attacks.

3) AUGMENTED REALITY

FL has even been proposed for use in Augmented Reality applications. The authors of [177] managed to discuss this in detail, discussing high data and latency issues for handling the enormous data for augmented reality. Their focus was mainly on solving these issues and they proposed a framework using FL. Specifically, their framework uses a combination of FL and mobile edge computing (MEC) in order to account for the huge amounts of data and latency issues. The authors' framework treats devices as a group, mainly in charge of generating data and performing local model updates. The cloud aspect of the framework facilitates the computation of the global model depending on local models that were uploaded. The authors tested their framework's performance with the CIFAR-10 dataset, which consists of ten thousand labeled images. Their framework resulted in requiring less training, which is impressive since training models can often take a lot of time.

XI. BEST PRACTICES IN DESIGNING FL MODELS

While FL has inspired many FL-based models and architectures, they each come with both advantages and drawbacks. Interestingly, because FL is still so new, it is a possibility that there are not any official guidelines on how to best design FL models [178], [179]. By establishing the best practices in designing FL models, the FL models that will be designed in the future will likely be better off in terms of efficiency, privacy, performance, and personalization [180]. Additionally, while the discussed FL models and architectures so far do guarantee privacy protection, a lot of the articles only mention either the benefits of privacy in FL or the drawbacks of incorporating FL to not compromise privacy [181]. Also, while it is true that data is often sensitive due to tight privacy and security laws, there has not been any mention of what exactly these regulations are and how they could directly impact the FL models' applicability; This is especially crucial for the medical setting since the data is so sensitive [182], [183]. As such, the best practices in designing FL models come from several aspects which are the following:

1) **Privacy** - As stated in this paper, privacy is the biggest aspect of FL. With privacy, we want to make sure that no sensitive data is being leaked.
2) **Integrity** - With this aspect, this deals with making sure there are no malicious models or components

in the FL learning process. If such malicious entities were present in the model design than the FL models could be corrupted and possibly used by hackers for malevolent purposes.

3) **Personalization** - Currently, there has been little attention on how to personalize FL models to better suit industry needs. So being able to personalize FL models will lead to better design and overall better applicability. How one industry designs FL models may be different for other industries.
4) **Security** - Going by the previous point, FL is not perfect. In fact, it is prone to many attacks, a few of which are mentioned by researchers. These attacks can severe consequences for the FL models since they could be fed false information, which can degrade the FL models and negatively impact industries as a whole. Thankfully, there has been research in combatting such security hazards when designing FL models.
5) **Efficiency and Effectiveness** - In designing the FL models, the models need to be both efficient and effective in order to obtain the most out of FL. That way the clients are able to manage their resources effectively [184].
6) **Complying with Privacy Regulations** - With this point, this is more about taking into consideration what current regulations exist regarding data sharing. This is especially true for healthcare since the data in the healthcare industry is very sensitive. There does not seem to be many works covering the legal ramifications of FL. These legalities regarding the current privacy regulations and how they work need to be researched so we can see how FL impacts these, and what changes (if any) might be needed to amend these current privacy regulations.

Overall, these aspects should be strongly considered when designing FL models. Hopefully, future works will also attempt to establish guidelines on the best practices for designing FL models as well [185]. By establishing guidelines on how the FL models should be designed, act, and perform, FL will be more applicable than it is currently.

XII. CONCLUSION

The emerging idea FL is rapidly finding its path throughout our modern life, aiming to improve security and handling of data to benefit many domains. Overall, FL would allow for seamless data sharing and data access while still being secure. This paper presented an overview of the premise of this concept, its enabling technologies, protocols, frameworks, and recent research addressing different aspects of FL. We also discussed several applications and use-cases of FL. This, in turn, should provide a good foundation for data scientists who are interested to gain an insight into FL technologies and protocols to understand the role of the different components and protocols that constitute FL. Furthermore, some of the benefits, challenges, and issues that pertain to the design and deployment of FL have been presented.

Scenarios, where FL would be most beneficial, are varied, and getting these added benefits from FL will require sustained research and development efforts before it can be properly streamlined [186]–[188]. The protocols and frameworks discussed attempt to alleviate at least some of FL's drawbacks, but systems heterogeneity is still a critical challenge for FL that still needs to be addressed. Various systems that use FL have their own software and hardware restrictions, which can impact performance, and highlights another issue that future frameworks and architectures need to address, fault tolerance. FL needs to address fault tolerance to ensure every device that participates in FL is accounted for, and ensure performance is not jeopardized. Additionally, there has not been enough discussion about how FL would impact current legislation regarding data acquisition and handling, specifically in regards to fairness. In many real-world contexts where privacy protection is desired, fairness is also desired. So it is important that future FL research is also focused on examining how FL might be able to address existing concerns about fairness in machine learning, and whether FL raises new fairness-related issues such as potential bias. Future directions and suggestions for industries that wish to incorporate FL should look into what parts of their infrastructure could need FL the most, and modify accordingly depending on their device set up. Network infrastructure should also be accounted for to ensure FL can operate as smoothly as possible.

ACKNOWLEDGMENT

The content is solely the responsibility of the authors and does not necessarily represent the official views of the National Institutes of Health. Any opinions, findings, and conclusions or recommendations expressed in this material are those of the author(s) and do not necessarily reflect the views of the National Science Foundation.

REFERENCES

[1] P. P. Liang, T. Liu, L. Ziyin, N. B. Allen, R. P. Auerbach, D. Brent, R. Salakhutdinov, and L.-P. Morency, "Think locally, act globally: Federated learning with local and global representations," 2020, *arXiv:2001.01523*. [Online]. Available: http://arxiv.org/abs/2001.01523
[2] H. H. Zhuo, W. Feng, Y. Lin, Q. Xu, and Q. Yang, "Federated deep reinforcement learning," Feb. 2020, *arXiv:1901.08277*. [Online]. Available: https://arxiv.org/abs/1901.08277
[3] H. Yu, Z. Liu, Y. Liu, T. Chen, M. Cong, X. Weng, D. Niyato, and Q. Yang, "A fairness-aware incentive scheme for federated learning," in *Proc. AAAI/ACM Conf. AI, Ethics, Soc.*, Feb. 2020, pp. 393–399.
[4] S. Truex, N. Baracaldo, A. Anwar, T. Steinke, H. Ludwig, R. Zhang, and Y. Zhou, "A hybrid approach to privacy-preserving federated learning," in *Proc. 12th ACM Workshop Artif. Intell. Secur. (AISec)*, 2019, pp. 1–11.
[5] A. A. Süzen and M. A. Simsek, "A novel approach to machine learning application to protection privacy data in healthcare: Federated learning," *Namık Kemal Tıp Dergisi*, vol. 8, no. 1, pp. 22–30, 2020.
[6] S. Lin, G. Yang, and J. Zhang, "Real-time edge intelligence in the making: A collaborative learning framework via federated meta-learning," 2020, *arXiv:2001.03229*. [Online]. Available: http://arxiv.org/abs/2001.03229
[7] S. R. Pandey, N. H. Tran, M. Bennis, Y. K. Tun, A. Manzoor, and C. S. Hong, "A crowdsourcing framework for on-device federated learning," *IEEE Trans. Wireless Commun.*, vol. 19, no. 5, pp. 3241–3256, May 2020.
[8] R. Shao, H. He, H. Liu, and D. Liu, "Stochastic channel-based federated learning for medical data privacy preserving," 2019, *arXiv:1910.11160*. [Online]. Available: http://arxiv.org/abs/1910.11160

[9] A. Alexander, A. Jiang, C. Ferreira, and D. Zurkiya, "An intelligent future for medical imaging: A market outlook on artificial intelligence for medical imaging," *J. Amer. College Radiol.*, vol. 17, no. 1, pp. 165–170, Jan. 2020.

[10] E. Bakopoulou, B. Tillman, and A. Markopoulou, "A federated learning approach for mobile packet classification," 2019, *arXiv:1907.13113*. [Online]. Available: http://arxiv.org/abs/1907.13113

[11] D. B. Larson, D. C. Magnus, M. P. Lungren, N. H. Shah, and C. P. Langlotz, "Ethics of using and sharing clinical imaging data for artificial intelligence: A proposed framework," *Radiology*, vol. 295, no. 3, 2020, Art. no. 192536.

[12] J. Kone ný, H. Brendan McMahan, D. Ramage, and P. Richtárik, "Federated optimization: Distributed machine learning for on-device intelligence," 2016, *arXiv:1610.02527*. [Online]. Available: http://arxiv.org/abs/1610.02527

[13] C. Ilias and S. Georgios, "Machine learning for all: A more robust federated learning framework," in *Proc. 5th Int. Conf. Inf. Syst. Secur. Privacy*, 2019, pp. 544–551.

[14] 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 *Proc. ACM SIGSAC Conf. Comput. Commun. Secur.* New York, NY, USA: Association for Computing Machinery, Oct. 2017, pp. 1175–1191, doi: 10.1145/3133956.3133982.

[15] P. Vepakomma, T. Swedish, R. Raskar, O. Gupta, and A. Dubey, "No peek: A survey of private distributed deep learning," 2018, *arXiv:1812.03288*. [Online]. Available: http://arxiv.org/abs/1812.03288

[16] T. Li, M. Sanjabi, A. Beirami, and V. Smith, "Fair resource allocation in federated learning," 2019, *arXiv:1905.10497*. [Online]. Available: http://arxiv.org/abs/1905.10497

[17] H. Ben Yedder, B. Cardoen, and G. Hamarneh, "Deep learning for biomedical image reconstruction: A survey," 2020, *arXiv:2002.12351*. [Online]. Available: http://arxiv.org/abs/2002.12351

[18] R. Doku, D. B. Rawat, and C. Liu, "Towards federated learning approach to determine data relevance in big data," in *Proc. IEEE 20th Int. Conf. Inf. Reuse Integr. for Data Sci. (IRI)*. Los Alamitos, CA, USA: IEEE Computer Society, Jul. 2019, pp. 184–192, doi: 10.1109/iri.2019.00039.

[19] A. Stoian, R. Ivan, I. Stoian, and A. Marichescu, "Current trends in medical imaging acquisition and communication," in *Proc. IEEE Int. Conf. Autom., Qual. Test., Robot.* vol. 3. IEEE, May 2008, pp. 94–99.

[20] F. Donovan. (Apr. 2019). *Federated Learning Balances Machine Learning With Patient Privacy*. Accessed: Apr. 8, 2020. [Online]. Available: https://hitinfrastructure.com/news/federated-learning-balances-machine-learning-with-patient-privacy

[21] H. Zhu, Z. Li, M. Cheah, and R. Siow Mong Goh, "Privacy-preserving weighted federated learning within oracle-aided MPC framework," 2020, *arXiv:2003.07630*. [Online]. Available: http://arxiv.org/abs/2003.07630

[22] A. Qayyum, J. Qadir, M. Bilal, and A. Al-Fuqaha, "Secure and robust machine learning for healthcare: A survey," 2020, *arXiv:2001.08103*. [Online]. Available: http://arxiv.org/abs/2001.08103

[23] S. R. Pfohl, A. M. Dai, and K. Heller, "Federated and differentially private learning for electronic health records," 2019, *arXiv:1911.05861*. [Online]. Available: http://arxiv.org/abs/1911.05861

[24] L. Liu, J. Zhang, S. H. Song, and K. B. Letaief, "Client-edge-cloud hierarchical federated learning," 2019, *arXiv:1905.06641*. [Online]. Available: https://arxiv.org/abs/1905.06641

[25] Y. Xia, "Watermarking federated deep neural network models," M.S. thesis, G2 Pro Gradu, Diplomityö, Turku, Finland, Mar. 2020. [Online]. Available: http://urn.fi/URN:NBN:fi:aalto-202003222594

[26] T. Ben-Nun and T. Hoefler, "Demystifying parallel and distributed deep learning: An in-depth concurrency analysis," *ACM Comput. Surv.*, vol. 52, no. 4, pp. 1–43, Sep. 2019.

[27] M. G. Poirot, P. Vepakomma, K. Chang, J. Kalpathy-Cramer, R. Gupta, and R. Raskar, "Split learning for collaborative deep learning in healthcare," 2019, *arXiv:1912.12115*. [Online]. Available: http://arxiv.org/abs/1912.12115

[28] P. Vepakomma, T. Swedish, R. Raskar, O. Gupta, and A. Dubey, "No peek: A survey of private distributed deep learning," 2018, *arXiv:1812.03288*. [Online]. Available: http://arxiv.org/abs/1812.03288

[29] P. Vepakomma, O. Gupta, T. Swedish, and R. Raskar, "Split learning for health: Distributed deep learning without sharing raw patient data," 2018, *arXiv:1812.00564*. [Online]. Available: http://arxiv.org/abs/1812.00564

[30] K. Hsieh, "Machine learning systems for highly-distributed and rapidly-growing data," 2019, *arXiv:1910.08663*. [Online]. Available: http://arxiv.org/abs/1910.08663

[31] Q. Li, Z. Wen, Z. Wu, S. Hu, N. Wang, and B. He, "A survey on federated learning systems: Vision, hype and reality for data privacy and protection," 2019, *arXiv:1907.09693*. [Online]. Available: http://arxiv.org/abs/1907.09693

[32] V. Kulkarni, M. Kulkarni, and A. Pant, "Survey of personalization techniques for federated learning," 2020, *arXiv:2003.08673*. [Online]. Available: https://arxiv.org/abs/2003.08673

[33] J. Geiping, H. Bauermeister, H. Dröge, and M. Moeller, "Inverting gradients—How easy is it to break privacy in federated learning?" 2020, *arXiv:2003.14053*. [Online]. Available: https://arxiv.org/abs/2003.14053

[34] Y. Liu, X. Yuan, Z. Xiong, J. Kang, X. Wang, and D. Niyato, "Federated learning for 6G communications: Challenges, methods, and future directions," 2020, *arXiv:2006.02931*. [Online]. Available: http://arxiv.org/abs/2006.02931

[35] K. Bonawitz, H. Eichner, W. Grieskamp, D. Huba, A. Ingerman, V. Ivanov, C. Kiddon, J. Konecný, S. Mazzocchi, H. B. McMahan, T. Van Overveldt, D. Petrou, D. Ramage, and J. Roselander, "Towards federated learning at scale: System design," 2019, *arXiv:1902.01046*. [Online]. Available: http://arxiv.org/abs/1902.01046

[36] K. Cheng, T. Fan, Y. Jin, Y. Liu, T. Chen, and Q. Yang, "SecureBoost: A lossless federated learning framework," 2019, *arXiv:1901.08755*. [Online]. Available: http://arxiv.org/abs/1901.08755

[37] Q. Yang, Y. Liu, T. Chen, and Y. Tong, "Federated machine learning: Concept and applications," *ACM Trans. Intell. Syst. Technol.*, vol. 10, no. 2, pp. 1–19, Feb. 2019, doi: 10.1145/3298981.

[38] Y. Liu, Y. Kang, C. Xing, T. Chen, and Q. Yang, "Secure federated transfer learning," 2018, *arXiv:1812.03337*. [Online]. Available: http://arxiv.org/abs/1812.03337

[39] Q. Jing, W. Wang, J. Zhang, H. Tian, and K. Chen, "Quantifying the performance of federated transfer learning," 2019, *arXiv:1912.12795*. [Online]. Available: http://arxiv.org/abs/1912.12795

[40] S. Caldas, V. Smith, and A. Talwalkar, "Federated kernelized multi-task learning," in *Proc. SysML Conf.*, 2018, pp. 1–3.

[41] S. Feng and H. Yu, "Multi-participant multi-class vertical federated learning," 2020, *arXiv:2001.11154*. [Online]. Available: http://arxiv.org/abs/2001.11154

[42] T.-D. Cao, T. Truong-Huu, H. Tran, and K. Tran, "A federated learning framework for privacy-preserving and parallel training," 2020, *arXiv:2001.09782*. [Online]. Available: http://arxiv.org/abs/2001.09782

[43] Q. Wu, K. He, and X. Chen, "Personalized federated learning for intelligent IoT applications: A cloud-edge based framework," 2020, *arXiv:2002.10671*. [Online]. Available: http://arxiv.org/abs/2002.10671

[44] A. Imteaj, U. Thakker, S. Wang, J. Li, and M. H. Amini, "Federated learning for resource-constrained iot devices: Panoramas and state-of-the-art," 2020, *arXiv:2002.10610*. [Online]. Available: https://arxiv.org/abs/2002.10610

[45] H.-K. Lim, J.-B. Kim, J.-S. Heo, and Y.-H. Han, "Federated reinforcement learning for training control policies on multiple IoT devices," *Sensors*, vol. 20, no. 5, p. 1359, Mar. 2020, doi: 10.3390/s20051359.

[46] B. Yin, H. Yin, Y. Wu, and Z. Jiang, "FDC: A secure federated deep learning mechanism for data collaborations in the Internet of Things," *IEEE Internet Things J.*, vol. 7, no. 7, pp. 6348–6359, Jul. 2020.

[47] E. Rizk, S. Vlaski, and A. H. Sayed, "Dynamic federated learning," 2020, *arXiv:2002.08782*. [Online]. Available: http://arxiv.org/abs/2002.08782

[48] O. Choudhury, A. Gkoulalas-Divanis, T. Salonidis, I. Sylla, Y. Park, G. Hsu, and A. Das, "Differential privacy-enabled federated learning for sensitive health data," 2019, *arXiv:1910.02578*. [Online]. Available: http://arxiv.org/abs/1910.02578

[49] Y. Chen, J. Wang, C. Yu, W. Gao, and X. Qin, "FedHealth: A federated transfer learning framework for wearable healthcare," 2019, *arXiv:1907.09173*. [Online]. Available: http://arxiv.org/abs/1907.09173

[50] D. Liu, T. Miller, R. Sayeed, and K. D. Mandl, "FADL:Federated-autonomous deep learning for distributed electronic health record," 2018, *arXiv:1811.11400*. [Online]. Available: http://arxiv.org/abs/1811.11400

[51] Y. Lu, X. Huang, Y. Dai, S. Maharjan, and Y. Zhang, "Blockchain and federated learning for privacy-preserved data sharing in industrial IoT," *IEEE Trans. Ind. Informat.*, vol. 16, no. 6, pp. 4177–4186, Jun. 2020.

[52] J. Xu and F. Wang, "Federated learning for healthcare informatics," 2019, *arXiv:1911.06270*. [Online]. Available: http://arxiv.org/abs/1911.06270
[53] Z. Chai, A. Ali, S. Zawad, S. Truex, A. Anwar, N. Baracaldo, Y. Zhou, H. Ludwig, F. Yan, and Y. Cheng, "TiFL: A tier-based federated learning system," 2020, *arXiv:2001.09249*. [Online]. Available: http://arxiv.org/abs/2001.09249
[54] 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 Things J.*, early access, Mar. 31, 2020, doi: 10.1109/JIOT.2020.2984332.
[55] R. Hu, Y. Gong, and Y. Guo, "CPFed: Communication-efficient and privacy-preserving federated learning," 2020, *arXiv:2003.13761*. [Online]. Available: http://arxiv.org/abs/2003.13761
[56] Y. Liu, S. Sun, Z. Ai, S. Zhang, Z. Liu, and H. Yu, "FedCoin: A peer-to-peer payment system for federated learning," 2020, *arXiv:2002.11711*. [Online]. Available: http://arxiv.org/abs/2002.11711
[57] Y. Li, C. Chen, N. Liu, H. Huang, Z. Zheng, and Q. Yan, "A blockchain-based decentralized federated learning framework with committee consensus," 2020, *arXiv:2004.00773*. [Online]. Available: http://arxiv.org/abs/2004.00773
[58] H. Kim, J. Park, M. Bennis, and S.-L. Kim, "Blockchained on-device federated learning," *IEEE Commun. Lett.*, vol. 24, no. 6, pp. 1279–1283, Jun. 2020.
[59] 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 Trans. Veh. Technol.*, vol. 69, no. 4, pp. 4298–4311, Apr. 2020.
[60] K. Toyoda and A. N. Zhang, "Mechanism design for an incentive-aware blockchain-enabled federated learning platform," in *Proc. IEEE Int. Conf. Big Data (Big Data)*, Los Angeles, CA, USA, 2019, pp. 395–403, doi: 10.1109/BigData47090.2019.9006344.
[61] G. Ulm, E. Gustavsson, and M. Jirstrand, "Functional federated learning in erlang (ffl-erl)," in *Proc. Int. Workshop Funct. Constraint Log. Program.* Cham, Switzerland: Springer, 2018, pp. 162–178.
[62] S. Caldas, S. M. K. Duddu, P. Wu, T. Li, J. Konecný, H. B. McMahan, V. Smith, and A. Talwalkar, "LEAF: A benchmark for federated settings," 2018, *arXiv:1812.01097*. [Online]. Available: http://arxiv.org/abs/1812.01097
[63] A. Nilsson, S. Smith, G. Ulm, E. Gustavsson, and M. Jirstrand, "A performance evaluation of federated learning algorithms," in *Proc. 2nd Workshop Distrib. Infrastructures Deep Learn. (DIDL)*, 2018, pp. 1–8.
[64] A. K. Sahu, T. Li, M. Sanjabi, M. Zaheer, A. Talwalkar, and V. Smith, "On the convergence of federated optimization in heterogeneous networks," Apr. 2020, *arXiv:1812.06127*. [Online]. Available: https://arxiv.org/abs/1812.06127
[65] H. Wang, M. Yurochkin, Y. Sun, D. Papailiopoulos, and Y. Khazaeni, "Federated learning with matched averaging," 2020, *arXiv:2002.06440*. [Online]. Available: http://arxiv.org/abs/2002.06440
[66] L. Huang, Y. Yin, Z. Fu, S. Zhang, H. Deng, and D. Liu, "LoAdaBoost:Loss-based AdaBoost federated machine learning on medical data," 2018, *arXiv:1811.12629*. [Online]. Available: http://arxiv.org/abs/1811.12629
[67] A. Guha Roy, S. Siddiqui, S. Pölsterl, N. Navab, and C. Wachinger, "BrainTorrent: A Peer-to-Peer environment for decentralized federated learning," 2019, *arXiv:1905.06731*. [Online]. Available: http://arxiv.org/abs/1905.06731
[68] Y. Deng, M. Mahdi Kamani, and M. Mahdavi, "Adaptive personalized federated learning," 2020, *arXiv:2003.13461*. [Online]. Available: http://arxiv.org/abs/2003.13461
[69] V. Felbab, P. Kiss, and T. Horváth, "Optimization in federated learning," in *Proc. 19th Conf. Inf. Technol.–Appl. Theory (ITAT)*, Donovaly, Slovakia, Sep. 2019, pp. 58–65.
[70] W. Liu, L. Chen, Y. Chen, and W. Zhang, "Accelerating federated learning via momentum gradient descent," *IEEE Trans. Parallel Distrib. Syst.*, vol. 31, no. 8, pp. 1754–1766, Aug. 2020.
[71] J. Konecny, J. Liu, P. Richtarik, and M. Takac, "Mini-batch semi-stochastic gradient descent in the proximal setting," *IEEE J. Sel. Topics Signal Process.*, vol. 10, no. 2, pp. 242–255, Mar. 2016.
[72] S. Ruder. (Jan. 2016). *An Overview of Gradient Descent Optimization Algorithms*. Accessed: Apr. 10, 2020. [Online]. Available: https://ruder.io/optimizing-gradient-descent/index.html
[73] C. Hansen. (Oct. 2019). *Optimizers Explained—Adam, Momentum and Stochastic Gradient Descent*. Accessed: Apr. 10, 2020. [Online]. Available: https://mlfromscratch.com/optimizers-explained/
[74] D. P. Kingma and J. Ba, "Adam: A method for stochastic optimization," in *Proc. 3rd Int. Conf. Learn. Represent. (ICLR)*, Y. Bengio and Y. LeCun, Eds. San Diego, CA, USA, May 2015. [Online]. Available: http://arxiv.org/abs/1412.6980
[75] S. Reddi, Z. Charles, M. Zaheer, Z. Garrett, K. Rush, J. Kone ný, S. Kumar, and H. Brendan McMahan, "Adaptive federated optimization," 2020, *arXiv:2003.00295*. [Online]. Available: http://arxiv.org/abs/2003.00295
[76] B. Brik, A. Ksentini, and M. Bouaziz, "Federated learning for UAVs-enabled wireless networks: Use cases, challenges, and open problems," *IEEE Access*, vol. 8, pp. 53841–53849, 2020.
[77] S. Wang, M. Chen, C. Yin, W. Saad, C. Seon Hong, S. Cui, and H. V. Poor, "Federated learning for task and resource allocation in wireless high altitude balloon networks," 2020, *arXiv:2003.09375*. [Online]. Available: http://arxiv.org/abs/2003.09375
[78] M. Chen, H. Vincent Poor, W. Saad, and S. Cui, "Convergence time optimization for federated learning over wireless networks," 2020, *arXiv:2001.07845*. [Online]. Available: http://arxiv.org/abs/2001.07845
[79] N. H. Tran, W. Bao, A. Zomaya, M. N. H. Nguyen, and C. S. Hong, "Federated learning over wireless networks: Optimization model design and analysis," in *Proc. IEEE Conf. Comput. Commun. (IEEE INFOCOM)*, Apr. 2019, pp. 1387–1395.
[80] M. Chen, Z. Yang, W. Saad, C. Yin, H. Poor, and S. Cui, "Performance optimization of federated learning over wireless networks," in *Proc. IEEE Global Commun. Conf. (GLOBECOM)*, Dec. 2019, pp. 1–6.
[81] K. Wei, J.-G. Li, M. Ding, C. Ma, H. Su, B. Zhang, and H. V. Poor, "Performance analysis and optimization in privacy-preserving federated learning," 2020, *arXiv:2003.00229*. [Online]. Available: https://arxiv.org/abs/2003.00229
[82] M. Chen, Z. Yang, W. Saad, C. Yin, H. Vincent Poor, and S. Cui, "A joint learning and communications framework for federated learning over wireless networks," 2019, *arXiv:1909.07972*. [Online]. Available: http://arxiv.org/abs/1909.07972
[83] H. H. Yang, Z. Liu, T. Q. S. Quek, and H. V. Poor, "Scheduling policies for federated learning in wireless networks," *IEEE Trans. Commun.*, vol. 68, no. 1, pp. 317–333, Jan. 2020.
[84] R. Fantacci and B. Picano, "Federated learning framework for mobile edge computing networks," *CAAI Trans. Intell. Technol.*, vol. 5, no. 1, pp. 15–21, Mar. 2020.
[85] S. Savazzi, M. Nicoli, and V. Rampa, "Federated learning with cooperating devices: A consensus approach for massive IoT networks," *IEEE Internet Things J.*, vol. 7, no. 5, pp. 4641–4654, May 2020.
[86] C. Zhang, "Online federated learning over decentralized networks," Ph.D. dissertation, Nanyang Technol. Univ., Singapore, 2018. [Online]. Available: https://hdl.handle.net/10356/106445
[87] J. Kang, Z. Xiong, D. Niyato, Y. Zou, Y. Zhang, and M. Guizani, "Reliable federated learning for mobile networks," 2019, *arXiv:1910.06837*. [Online]. Available: http://arxiv.org/abs/1910.06837
[88] S. Li, Y. Cheng, W. Wang, Y. Liu, and T. Chen, "Learning to detect malicious clients for robust federated learning," 2020, *arXiv:2002.00211*. [Online]. Available: http://arxiv.org/abs/2002.00211
[89] T. T. Vu, D. T. Ngo, N. H. Tran, H. Quoc Ngo, M. N. Dao, and R. H. Middleton, "Cell-free massive MIMO for wireless federated learning," 2019, *arXiv:1909.12567*. [Online]. Available: http://arxiv.org/abs/1909.12567
[90] B. Liu, L. Wang, and M. Liu, "Lifelong federated reinforcement learning: A learning architecture for navigation in cloud robotic systems," *IEEE Robot. Autom. Lett.*, vol. 4, no. 4, pp. 4555–4562, Oct. 2019.
[91] O. Choudhury, A. Gkoulalas-Divanis, T. Salonidis, I. Sylla, Y. Park, G. Hsu, and A. Das, "Anonymizing data for privacy-preserving federated learning," 2020, *arXiv:2002.09096*. [Online]. Available: http://arxiv.org/abs/2002.09096
[92] K. D. Mandl and I. S. Kohane, "Federalist principles for healthcare data networks," *Nature Biotechnol.*, vol. 33, no. 4, p. 360, 2015.
[93] M. Hao, H. Li, X. Luo, G. Xu, H. Yang, and S. Liu, "Efficient and privacy-enhanced federated learning for industrial artificial intelligence," *IEEE Trans. Ind. Informat.*, vol. 16, no. 10, pp. 6532–6542, Oct. 2020.
[94] N. Yoshida, T. Nishio, M. Morikura, K. Yamamoto, and R. Yonetani, "Hybrid-FL for wireless networks: Cooperative learning mechanism using non-IID data," 2019, *arXiv:1905.07210*. [Online]. Available: http://arxiv.org/abs/1905.07210
[95] R. Xu, N. Baracaldo, Y. Zhou, A. Anwar, and H. Ludwig, "HybridAlpha: An efficient approach for privacy-preserving federated learning," in *Proc. 12th ACM Workshop Artif. Intell. Secur. (AISec)*, 2019, pp. 13–23.

[96] T. Nishio and R. Yonetani, "Client selection for federated learning with heterogeneous resources in mobile edge," in *Proc. IEEE Int. Conf. Commun. (ICC)*, May 2019, pp. 1–7.
[97] K. Mandal and G. Gong, "PrivFL: Practical privacy-preserving federated regressions on high-dimensional data over mobile networks," in *Proc. CCSW*, 2019, pp. 57–68.
[98] G. Xu, H. Li, S. Liu, K. Yang, and X. Lin, "VerifyNet: Secure and verifiable federated learning," *IEEE Trans. Inf. Forensics Security*, vol. 15, pp. 911–926, 2020.
[99] Y. Liu, J. J. Q. Yu, J. Kang, D. Niyato, and S. Zhang, "Privacy-preserving traffic flow prediction: A federated learning approach," 2020, *arXiv:2003.08725*. [Online]. Available: http://arxiv.org/abs/2003.08725
[100] M. Nasr, R. Shokri, and A. Houmansadr, "Comprehensive privacy analysis of deep learning," in *Proc. IEEE Symp. Secur. Privacy (SP)*, May 2019, pp. 739–753.
[101] Y. Qu, L. Gao, T. H. Luan, Y. Xiang, S. Yu, B. Li, and G. Zheng, "Decentralized privacy using blockchain-enabled federated learning in fog computing," *IEEE Internet Things J.*, vol. 7, no. 6, pp. 5171–5183, Jun. 2020.
[102] P. Kairouz *et al.*, "Advances and open problems in federated learning," 2019, *arXiv:1912.04977*. [Online]. Available: http://arxiv.org/abs/1912.04977
[103] Y. Zhao, M. Li, L. Lai, N. Suda, D. Civin, and V. Chandra, "Federated learning with non-IID data," 2018, *arXiv:1806.00582*. [Online]. Available: http://arxiv.org/abs/1806.00582
[104] A. Albaseer, B. Sait Ciftler, M. Abdallah, and A. Al-Fuqaha, "Exploiting unlabeled data in smart cities using federated learning," 2020, *arXiv:2001.04030*. [Online]. Available: http://arxiv.org/abs/2001.04030
[105] N. Balachandar, K. Chang, J. Kalpathy-Cramer, and D. L. Rubin, "Accounting for data variability in multi-institutional distributed deep learning for medical imaging," *J. Amer. Med. Inform. Assoc.*, vol. 27, no. 5, pp. 700–708, May 2020.
[106] D. Verma, G. White, S. Julier, S. Pasteris, S. Chakraborty, and G. Cirincione, "Approaches to address the data skew problem in federated learning," in *Proc. Artif. Intell. Mach. Learn. Multi-Domain Operations Appl.*, May 2019, Art. no. 110061.
[107] J. Qian, X. Fafoutis, and L. Kai Hansen, "Towards federated learning: Robustness analytics to data heterogeneity," 2020, *arXiv:2002.05038*. [Online]. Available: http://arxiv.org/abs/2002.05038
[108] N. Aussel, S. Chabridon, and Y. Petetin, "Combining federated and active learning for communication-efficient distributed failure prediction in aeronautics," 2020, *arXiv:2001.07504*. [Online]. Available: http://arxiv.org/abs/2001.07504
[109] S. Samarakoon, M. Bennis, W. Saad, and M. Debbah, "Federated learning for ultra-reliable low-latency v2v communications," in *Proc. GLOBECOM*, Dec. 2018, pp. 1–7.
[110] L. Wang, W. Wang, and B. Li, "CMFL: Mitigating communication overhead for federated learning," in *Proc. IEEE 39th Int. Conf. Distrib. Comput. Syst. (ICDCS)*, Jul. 2019, pp. 954–964.
[111] F. Chen, M. Luo, Z. Dong, Z. Li, and X. He, "Federated meta-learning with fast convergence and efficient communication," 2018, *arXiv:1802.07876*. [Online]. Available: https://arxiv.org/abs/1802.07876
[112] T. Li, A. Kumar Sahu, A. Talwalkar, and V. Smith, "Federated learning: Challenges, methods, and future directions," 2019, *arXiv:1908.07873*. [Online]. Available: http://arxiv.org/abs/1908.07873
[113] S. Ji, W. Jiang, A. Walid, and X. Li, "Dynamic sampling and selective masking for communication-efficient federated learning," 2020, *arXiv:2003.09603*. [Online]. Available: http://arxiv.org/abs/2003.09603
[114] B. Liu, L. Wang, M. Liu, and C.-Z. Xu, "Federated imitation learning: A privacy considered imitation learning framework for cloud robotic systems with heterogeneous sensor data," 2019, *arXiv:1909.00895*. [Online]. Available: http://arxiv.org/abs/1909.00895
[115] B. Sait Ciftler, A. Albaseer, N. Lasla, and M. Abdallah, "Federated learning for localization: A privacy-preserving crowdsourcing method," 2020, *arXiv:2001.01911*. [Online]. Available: http://arxiv.org/abs/2001.01911
[116] J. Kone ný, H. Brendan McMahan, F. X. Yu, P. Richtárik, A. Theertha Suresh, and D. Bacon, "Federated learning: Strategies for improving communication efficiency," 2016, *arXiv:1610.05492*. [Online]. Available: http://arxiv.org/abs/1610.05492
[117] M. Manimel Wadu, S. Samarakoon, and M. Bennis, "Federated learning under channel uncertainty: Joint client scheduling and resource allocation," 2020, *arXiv:2002.00802*. [Online]. Available: http://arxiv.org/abs/2002.00802
[118] F. Ang, L. Chen, N. Zhao, Y. Chen, W. Wang, and F. R. Yu, "Robust federated learning with noisy communication," 2019, *arXiv:1911.00251*. [Online]. Available: http://arxiv.org/abs/1911.00251
[119] X. Yao, C. Huang, and L. Sun, "Two-stream federated learning: Reduce the communication costs," in *Proc. IEEE Vis. Commun. Image Process. (VCIP)*, Dec. 2018, pp. 1–4.
[120] S. Niknam, H. S. Dhillon, and J. H. Reed, "Federated learning for wireless communications: Motivation, opportunities and challenges," 2019, *arXiv:1908.06847*. [Online]. Available: http://arxiv.org/abs/1908.06847
[121] W. Yang Bryan Lim, N. Cong Luong, D. Thai Hoang, Y. Jiao, Y.-C. Liang, Q. Yang, D. Niyato, and C. Miao, "Federated learning in mobile edge networks: A comprehensive survey," 2019, *arXiv:1909.11875*. [Online]. Available: http://arxiv.org/abs/1909.11875
[122] C. Ma, J. Li, M. Ding, H. Hao Yang, F. Shu, T. Q. S. Quek, and H. V. Poor, "On safeguarding privacy and security in the framework of federated learning," 2019, *arXiv:1909.06512*. [Online]. Available: http://arxiv.org/abs/1909.06512
[123] F. Nuding and R. Mayer, "Poisoning attacks in federated learning: An evaluation on traffic sign classification," in *Proc. 10th ACM Conf. Data Appl. Secur. Privacy*, 2020, pp. 168–170.
[124] Y. Feng, X. Yang, W. Fang, S.-T. Xia, and X. Tang, "Practical and bilateral privacy-preserving federated learning," 2020, *arXiv:2002.09843*. [Online]. Available: http://arxiv.org/abs/2002.09843
[125] K. Yang, T. Jiang, Y. Shi, and Z. Ding, "Federated learning based on over-the-air computation," in *Proc. ICC - IEEE Int. Conf. Commun. (ICC)*, May 2019, pp. 1–6.
[126] N. Rieke, J. Hancox, W. Li, F. Milletari, H. Roth, S. Albarqouni, S. Bakas, M. N. Galtier, B. Landman, K. Maier-Hein, S. Ourselin, M. Sheller, R. M. Summers, A. Trask, D. Xu, M. Baust, and M. J. Cardoso, "The future of digital health with federated learning," 2020, *arXiv:2003.08119*. [Online]. Available: http://arxiv.org/abs/2003.08119
[127] Z. Li, V. Sharma, and S. P. Mohanty, "Preserving data privacy via federated learning: Challenges and solutions," *IEEE Consum. Electron. Mag.*, vol. 9, no. 3, pp. 8–16, May 2020.
[128] W. Du, X. Zeng, M. Yan, and M. Zhang, "Efficient federated learning via variational dropout," openreview.net, Dec. 2018, pp. 1–12. [Online]. Available: https://openreview.net/forum?id=BkeAf2CqY7
[129] W. Raghupathi and V. Raghupathi, "Big data analytics in healthcare: Promise and potential," *Health Inf. Sci. Syst.*, vol. 2, no. 1, p. 3, Dec. 2014.
[130] Y. Laguel, K. Pillutla, J. Malick, and Z. Harchaoui, "Device heterogeneity in federated learning: A superquantile approach," 2020, *arXiv:2002.11223*. [Online]. Available: http://arxiv.org/abs/2002.11223
[131] S.-J. Hahn and J. Lee, "Privacy-preserving federated Bayesian learning of a generative model for imbalanced classification of clinical data," 2019, *arXiv:1910.08489*. [Online]. Available: https://arxiv.org/abs/1910.08489
[132] S. Dash, S. K. Shakyawar, M. Sharma, and S. Kaushik, "Big data in healthcare: Management, analysis and future prospects," *J. Big Data*, vol. 6, no. 1, p. 54, Dec. 2019.
[133] Y. Chen, X. Sun, and Y. Jin, "Communication-efficient federated deep learning with asynchronous model update and temporally weighted aggregation," 2019, *arXiv:1903.07424*. [Online]. Available: http://arxiv.org/abs/1903.07424
[134] M. U. Asad, A. A. Moustafa, T. Ito, and M. Aslam, "Evaluating the communication efficiency in federated learning algorithms," 2020, *arXiv:2004.02738*. [Online]. Available: https://arxiv.org/abs/2004.02738
[135] R. Rodriguez, G. Svensson, and G. Wood, "Sustainability trends in public hospitals: Efforts and priorities," *Eval. Program Planning*, vol. 78, Feb. 2020, Art. no. 101742. [Online]. Available: http://www.sciencedirect.com/science/article/pii/S0149718919303465
[136] (Jan. 2020). *Big Data Spending in Healthcare Market 2020 Global Industry Size, Share, Revenue Growth Development, Business Opportunities, Future Trends, Top Key Players, and Global Analysis by Forecast to 2025—Marketwatch*. Accessed: Aug. 8, 2020. [Online]. Available: https://www.marketwatch.com/press-release/big-data-spending-inhealthcare-market-2020-global-industry-size-share-revenue-growthdevelopment-business-opportunities-future-trends-top-key-playersand-global-analysis-by-forecast-to-2025-2020-01-27
[137] (Jun. 2018). *National Health Expenditure Projections 2018–2027*. Accessed: Apr. 8, 2020. [Online]. Available: https://www.cms.gov/Research-Statistics-Data-and-Systems/Statistics-Trends-and-Reports/NationalHealthExpendData/Downloads/ForecastSummary.pdf
[138] J. Kent. (Dec. 2018). *Big Data to See Explosive Growth, Challenging Healthcare Organizations*. Accessed: Apr. 8, 2020. [Online]. Available: https://healthitanalytics.com/news/big-data-to-see-explosive-growth-chalenging-healthcare-organizations

[139] T. Joda, M. M. Bornstein, R. E. Jung, M. Ferrari, T. Waltimo, and N. U. Zitzmann, "Recent trends and future direction of dental research in the digital era," *Int. J. Environ. Res. Public Health*, vol. 17, no. 6, p. 1987, Mar. 2020.

[140] A. More. (Feb. 2020). *Big Data Spending in Healthcare Market 2020 Byglobal Share, Size, Industry Dynamics, Manufacturers, Type and Application, Forecast to 2025—Marketwatch*. Accessed: Apr. 8, 2020. [Online]. Available: https://www.marketwatch.com/press-release/big-data-spending-in-healthcare-market-2020-byglobal-share-size-industry-dynamics-manufacturers-type-and-application-forecast-to-2025-2020-02-03

[141] (Mar. 2018). *Healthcare Analytics Market Size Worth $53.65 Billion by 2025*. Accessed: Apr. 8, 2020. [Online]. Available: https://www.grandviewresearch.com/press-release/global-healthcare-analytics-market

[142] F. R. Vogenberg, "Us healthcare trends and contradictions in 2019," *Am Health Drug Benefits*, vol. 12, no. 1, pp. 40–47, Feb. 2019. Accessed: Apr. 8, 2020.

[143] M. Mikulic. (2020). *Health Big Data Market Size 2025 Global Forecast | Statista*. Accessed: Apr. 8, 2020. [Online]. Available: https://www.statista.com/statistics/909654/global-big-data-in-healthcare-market-size/

[144] M. Burger, "The risk to population health equity posed by automated decision systems: A narrative review," 2020, *arXiv:2001.06615*. [Online]. Available: http://arxiv.org/abs/2001.06615

[145] M. Seif, R. Tandon, and M. Li, "Wireless federated learning with local differential privacy," 2020, *arXiv:2002.05151*. [Online]. Available: http://arxiv.org/abs/2002.05151

[146] L. Lyu, H. Yu, and Q. Yang, "Threats to federated learning: A survey," 2020, *arXiv:2003.02133*. [Online]. Available: https://arxiv.org/abs/2003.02133

[147] Y. Liu, Z. Ma, X. Liu, S. Ma, S. Nepal, and R. Deng, "Boosting privately: Privacy-preserving federated extreme boosting for mobile crowdsensing," 2019, *arXiv:1907.10218*. [Online]. Available: http://arxiv.org/abs/1907.10218

[148] X. Yao, T. Huang, C. Wu, R. Zhang, and L. Sun, "Towards faster and better federated learning: A feature fusion approach," in *Proc. IEEE Int. Conf. Image Process. (ICIP)*, Sep. 2019, pp. 175–179.

[149] Y. Mansour, M. Mohri, J. Ro, and A. Theertha Suresh, "Three approaches for personalization with applications to federated learning," 2020, *arXiv:2002.10619*. [Online]. Available: http://arxiv.org/abs/2002.10619

[150] Y. Liu, Z. Ma, X. Liu, and J. Ma, "Learn to forget: User-level memorization elimination in federated learning," 2020, *arXiv:2003.10933*. [Online]. Available: http://arxiv.org/abs/2003.10933

[151] J. So, B. Guler, and A. S. Avestimehr, "Turbo-aggregate: Breaking the quadratic aggregation barrier in secure federated learning," *IACR Cryptol. ePrint Arch.*, vol. 2020, p. 167, Feb. 2020.

[152] T. Yu, E. Bagdasaryan, and V. Shmatikov, "Salvaging federated learning by local adaptation," 2020, *arXiv:2002.04758*. [Online]. Available: http://arxiv.org/abs/2002.04758

[153] H. Zhu and Y. Jin, "Real-time federated evolutionary neural architecture search," 2020, *arXiv:2003.02793*. [Online]. Available: http://arxiv.org/abs/2003.02793

[154] T. Yang, G. Andrew, H. Eichner, H. Sun, W. Li, N. Kong, D. Ramage, and F. Beaufays, "Applied federated learning: Improving Google keyboard query suggestions," 2018, *arXiv:1812.02903*. [Online]. Available: http://arxiv.org/abs/1812.02903

[155] D. Leroy, A. Coucke, T. Lavril, T. Gisselbrecht, and J. Dureau, "Federated learning for keyword spotting," in *Proc. IEEE Int. Conf. Acoust., Speech Signal Process. (ICASSP)*, May 2019, pp. 6341–6345.

[156] A. Hard, K. Rao, R. Mathews, S. Ramaswamy, F. Beaufays, S. Augenstein, H. Eichner, C. Kiddon, and D. Ramage, "Federated learning for mobile keyboard prediction," 2018, *arXiv:1811.03604*. [Online]. Available: http://arxiv.org/abs/1811.03604

[157] F. Hartmann, S. Suh, A. Komarzewski, T. D. Smith, and I. Segall, "Federated learning for ranking browser history suggestions," 2019, *arXiv:1911.11807*. [Online]. Available: http://arxiv.org/abs/1911.11807

[158] Y. Liu, A. Huang, Y. Luo, H. Huang, Y. Liu, Y. Chen, L. Feng, T. Chen, H. Yu, and Q. Yang, "FedVision: An online visual object detection platform powered by federated learning," 2020, *arXiv:2001.06202*. [Online]. Available: http://arxiv.org/abs/2001.06202

[159] K. Sozinov, V. Vlassov, and S. Girdzijauskas, "Human activity recognition using federated learning," in *Proc. IEEE Intl Conf Parallel Distrib. Process. with Appl., Ubiquitous Comput. Commun., Big Data Cloud Comput., Social Comput. Netw., Sustain. Comput. Commun. (ISPA/IUCC/BDCloud/SocialCom/SustainCom)*, Dec. 2018, pp. 1103–1111.

[160] L. Huang and D. Liu, "Patient clustering improves efficiency of federated machine learning to predict mortality and hospital stay time using distributed electronic medical records," *J. Biomed. Informat.*, vol. 99, p. 21, Nov. 2019.

[161] Z. Xiong, Z. Cheng, X. Liu, D. Wang, X. Luo, K. Chen, H. Jiang, and M. Zheng, "Facing small and biased data dilemma in drug discovery with federated learning," *BioRxiv*, 2020. [Online]. Available: https://www.biorxiv.org/content/early/2020/03/20/2020.03.19.998898

[162] S. Chen, D. Xue, G. Chuai, Q. Yang, and Q. Liu. (2020). FL-QSAR: A Federated Learning Based QSAR Prototype for Collaborative Drug Discovery. Biorxiv. [Online]. Available: https://www.biorxiv.org/content/early/2020/02/28/2020.02.27.950592

[163] X. Li, Y. Gu, N. Dvornek, L. Staib, P. Ventola, and J. S. Duncan, "Multi-site fMRI analysis using privacy-preserving federated learning and domain adaptation: ABIDE results," 2020, *arXiv:2001.05647*. [Online]. Available: http://arxiv.org/abs/2001.05647

[164] S. Li, Y. Cheng, Y. Liu, W. Wang, and T. Chen, "Abnormal client behavior detection in federated learning," 2019, *arXiv:1910.09933*. [Online]. Available: http://arxiv.org/abs/1910.09933

[165] M. J. Sheller, G. A. Reina, B. Edwards, J. Martin, and S. Bakas, "Multi-institutional deep learning modeling without sharing patient data: A feasibility study on brain tumor segmentation," in *Proc. Int. MICCAI Brainlesion Workshop*, in Lecture Notes in Computer Science, vol. 11383. Cham, Switzerland: Springer, Jan. 2019, pp. 92–104, doi: 10.1007/978-3-030-11723-8_9.

[166] W. Li, F. Milletari, D. Xu, N. Rieke, J. Hancox, W. Zhu, M. Baust, Y. Cheng, S. Ourselin, M. J. Cardoso, and A. W. Feng, "Privacy-preserving federated brain tumour segmentation," in *Proc. MLMI@MICCAI*, 2019, pp. 133–141.

[167] S. Silva, B. A. Gutman, E. Romero, P. M. Thompson, A. Altmann, and M. Lorenzi, "Federated learning in distributed medical databases: Meta-analysis of large-scale subcortical brain data," in *Proc. IEEE 16th Int. Symp. Biomed. Imag. (ISBI)*, Apr. 2019, pp. 270–274.

[168] H. Sharghi, W. Ma, and K. Sartipi, "Federated service-based authentication provisioning for distributed diagnostic imaging systems," in *Proc. IEEE 28th Int. Symp. Comput.-Based Med. Syst.*, Jun. 2015, pp. 344–347.

[169] S. Ge, F. Wu, C. Wu, T. Qi, Y. Huang, and X. Xie, "FedNER: Privacy-preserving medical named entity recognition with federated learning," 2020, *arXiv:2003.09288*. [Online]. Available: http://arxiv.org/abs/2003.09288

[170] T. Qi, F. Wu, C. Wu, Y. Huang, and X. Xie, "FedRec: Privacy-preserving news recommendation with federated learning," 2020, *arXiv:2003.09592*. [Online]. Available: https://arxiv.org/abs/2003.09592

[171] F. Yin, Z. Lin, Y. Xu, Q. Kong, D. Li, S. Theodoridis, and S. Cui, "FedLoc: Federated learning framework for data-driven cooperative localization and location data processing," 2020, *rXiv:2003.03697*. [Online]. Available: https://arxiv.org/abs/2003.03697

[172] R. Zeng, S. Zhang, J. Wang, and X. Chu, "FMore: An incentive scheme of multi-dimensional auction for federated learning in MEC," 2020, *arXiv:2002.09699*. [Online]. Available: http://arxiv.org/abs/2002.09699

[173] Y. Chen, X. Yang, X. Qin, H. Yu, B. Chen, and Z. Shen, "FOCUS: Dealing with label quality disparity in federated learning," 2020, *arXiv:2001.11359*. [Online]. Available: http://arxiv.org/abs/2001.11359

[174] Y. Jiang, J. Kone ný, K. Rush, and S. Kannan, "Improving federated learning personalization via model agnostic meta learning," 2019, *arXiv:1909.12488*. [Online]. Available: http://arxiv.org/abs/1909.12488

[175] I. Blanquer, Á. Alberich-Bayarri, F. García-Castro, G. Teodoro, A. Meirelles, B. Nascimento, W. Meira, Jr., and A. Ribeiro, "Medical imaging processing architecture on ATMOSPHERE federated platform," in *Proc. 9th Int. Conf. Cloud Comput. Services Sci.*, Jan. 2019, pp. 589–594.

[176] T. D. Nguyen, S. Marchal, M. Miettinen, H. Fereidooni, N. Asokan, and A.-R. Sadeghi, "DÏoT: A federated self-learning anomaly detection system for IoT," in *Proc. IEEE 39th Int. Conf. Distrib. Comput. Syst. (ICDCS)*, May 2019, pp. 756–767.

[177] D. Chen, L. J. Xie, B. Kim, L. Wang, C. S. Hong, L.-C. Wang, and Z. Han, "Federated learning based mobile edge computing for augmented reality applications," in *Proc. Int. Conf. Comput., Netw. Commun. (ICNC)*, Feb. 2020, pp. 767–773.

[178] J. Kone ný, H. B. McMahan, and D. Ramage, "Federated optimization: Distributed optimization beyond the datacenter," 2015, *arXiv:1511.03575*. [Online]. Available: https://arxiv.org/abs/1511.03575
[179] R. Nock, S. Hardy, W. Henecka, H. Ivey-Law, G. Patrini, G. Smith, and B. Thorne, "Entity resolution and federated learning get a federated resolution," 2018, *arXiv:1803.04035*. [Online]. Available: http://arxiv.org/abs/1803.04035
[180] A. Fallah, A. Mokhtari, and A. Ozdaglar, "Personalized federated learning: A meta-learning approach," 2020, *arXiv:2002.07948*. [Online]. Available: http://arxiv.org/abs/2002.07948
[181] Y. Huang, "Preservation of patient level privacy: Federated classification and calibration models," Ph.D. dissertation, Univ. California San Diego, San Diego, CA, USA, 2020. [Online]. Available: https://escholarship.org/uc/item/9dr8s83t
[182] 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," 2016, *arXiv:1611.04482*. [Online]. Available: http://arxiv.org/abs/1611.04482
[183] X. Lu, Y. Liao, P. Lio, and P. Hui, "Privacy-preserving asynchronous federated learning mechanism for edge network computing," *IEEE Access*, vol. 8, pp. 48970–48981, 2020.
[184] S. Caldas, J. Kone ny, H. Brendan McMahan, and A. Talwalkar, "Expanding the reach of federated learning by reducing client resource requirements," 2018, *arXiv:1812.07210*. [Online]. Available: http://arxiv.org/abs/1812.07210
[185] F. Hanzely and P. Richtárik, "Federated learning of a mixture of global and local models," 2020, *arXiv:2002.05516*. [Online]. Available: http://arxiv.org/abs/2002.05516
[186] A. Alexander, M. McGill, A. Tarasova, C. Ferreira, and D. Zurkiya, "Scanning the future of medical imaging," *J. Amer. College Radiol.*, vol. 16, no. 4, pp. 501–507, Apr. 2019. [Online]. Available: http://www.sciencedirect.com/science/article/pii/S1546144018312821
[187] R. Agarwal, G. G. Gao, C. DesRoches, and A. K. Jha, "Research commentary—The digital transformation of healthcare: Current status and the road ahead," *Inf. Syst. Res.*, vol. 21, no. 4, pp. 796–809, Dec. 2010, doi: 10.1287/isre.1100.0327.
[188] M. Jones, F. DeRuyter, and J. Morris, "The digital health revolution and people with disabilities: Perspective from the united states," *Int. J. Environ. Res. Public Health*, vol. 17, no. 2, p. 381, Jan. 2020, doi: 10.3390/ijerph17020381.

MOHAMMED ALEDHARI (Member, IEEE) received the Ph.D. degree from the Department of Computer Science, Western Michigan University, in December 2017. He is currently an Assistant Professor of computer science at Kennesaw State University, and is also the Director of the Smart and Autonomous Systems Center (SASC), KSU, since January 2019. His research interests include data science, machine learning, computer vision for medical imaging and autonomous systems, and big data problems in computational biology and bioinformatics. Prior to joining KSU, he was a Postdoctoral Fellow at the Center for High Performance Computing and Big Data (CHPCBD) of Computer Science, Western Michigan University (WMU), Kalamazoo, MI, USA, from January 2018 to August 2018. Prior to joining CHPCBD, he served as a Graduate Research Assistant at the Computer Networks, Embedded Systems and Telecommunications (NEST) Laboratory. He was also a Doctoral Associate with the Graduate Research and Retention, Western Michigan University (WMU), Kalamazoo. He is a member of the Institute of Electrical and Electronics Engineers (IEEE), Association for Computing Machinery (ACM), and American Society for Quality (ASQ). His honors include All-University Graduate Research and Creative Scholar Award, in 2018, nominated to the Early Achievement Award of the IEEE Communications Society, in 2019, and the Gwen Frostic Doctoral Fellowship, in 2016.

REHMA RAZZAK received the B.S. degree in computer science from Kennesaw State University, GA, USA, in 2018. She is currently pursuing the M.S. degree in computer science from Kennesaw State University. She also has plans to pursue her Ph.D. degree in data science. She is currently a Graduate Research Assistant at the Smart and Autonomous Systems Center (SASC), College of Computing and Software Engineering, Kennesaw State University. Two of her research projects earned nominations for the Best Research Project, in fall 2018. Her current research interests include machine learning, game development, and autism spectrum disorder (ASD).

REZA M. PARIZI (Senior Member, IEEE) received the B.Sc. and M.Sc. degrees in computer science in 2005 and 2008, respectively, and the Ph.D. degree in software engineering in 2012. He is currently the Director of the Decentralized Science Lab (dSL), Kennesaw State University, GA, USA. He is a Consummate AI Technologist and Software Security Researcher with an entrepreneurial spirit. He is a senior member of the IEEE Blockchain Community and ACM. Prior to joining KSU, he was with the New York Institute of Technology. His research interests include Research and Development in decentralized AI, cybersecurity, blockchain systems, smart contracts, and emerging issues in the practice of secure software-run world applications.

FAHAD SAEED (Senior Member, IEEE) received the Ph.D. degree from the Department of Electrical and Computer Engineering, University of Illinois at Chicago (UIC), in 2010. He is currently an Associate Professor at the School of Computing and Information Sciences, Florida International University (FIU), Miami, FL, USA, and is the Director of the Saeed Lab, FIU. Prior to joining FIU, he was a tenure-track Assistant Professor with the Department of Electrical and Computer Engineering and the Department of Computer Science, Western Michigan University (WMU), Kalamazoo, MI, USA, since January 2014. He was tenured and promoted to the rank of Associate Professor at WMU, in August 2018. He was a Postdoctoral Fellow and then a Research Fellow with the Systems Biology Center, National Institutes of Health (NIH), Bethesda, MD, USA, from August 2010 to June 2011 and from June 2011 to January 2014, respectively. He has served as a visiting scientist in world-renowned prestigious institutions, such as the Department of Biosystems science and Engineering (D-BSSE), ETH Zürich, the Swiss Institute of Bioinformatics (SIB), and the Epithelial Systems Biology Laboratory (ESBL), National Institutes of Health (NIH), Bethesda, MD, USA. His research interests include parallel and distributed algorithms and architectures, computational proteomics and genomics, and big data problems in computational biology and bioinformatics. He is a Senior Member of ACM. His honors include the ThinkSwiss Fellowship, in 2007 and 2008, the NIH Postdoctoral Fellowship Award, in 2010, the Fellows Award for Research Excellence (FARE) at NIH, in 2012, the NSF CRII Award, in 2015, the WMU Outstanding New Researcher Award, in 2016, the WMU Distinguished Research and Creative Scholarship Award, in 2018, and the NSF CAREER Award, in 2017.