applied sciences

MDPI

Article

Secure Smart Communication Efficiency in Federated Learning: Achievements and Challenges

Seyedamin Pouriyeh [1], Osama Shahid [1], Reza M. Parizi [2], Quan Z. Sheng [3], Gautam Srivastava [4,5,6,*], Liang Zhao [1] and Mohammad Nasajpour [1]

[1] Department of Information Technology, Kennesaw State University, Marietta, GA 30060, USA
[2] Department of Software Engineering and Game Development, Kennesaw State University, Marietta, GA 30060, USA
[3] Department of Computing, Macquarie University, Sydney, NSW 2109, Australia
[4] Department of Math and Computer Science, Brandon University, Brandon, MB R7A 6A9, Canada
[5] Research Center for Interneural Computing, China Medical University, Taichung 40402, Taiwan
[6] Department of Computer Science and Mathematics, Lebanese American University, Beirut 1102, Lebanon
* Correspondence: srivastavag@brandonu.ca

Abstract: Federated learning (FL) is known to perform machine learning tasks in a distributed manner. Over the years, this has become an emerging technology, especially with various data protection and privacy policies being imposed. FL allows for performing machine learning tasks while adhering to these challenges. As with the emergence of any new technology, there will be challenges and benefits. A challenge that exists in FL is the communication costs: as FL takes place in a distributed environment where devices connected over the network have to constantly share their updates, this can create a communication bottleneck. This paper presents the state-of-the-art of the conducted works on communication constraints of FL while maintaining the secure and smart properties that federated learning is known for. Overall, current challenges and possible methods for enhancing the FL models' efficiency with a perspective on communication are discussed. This paper aims to bridge the gap in all conducted review papers by solely focusing on communication aspects in FL environments.

Keywords: security; privacy; smart environments; federated learning; communication; machine learning

Citation: Pouriyeh, S.; Shahid, O.; Parizi, R.M.; Sheng, Q.Z.; Srivastava, G.; Zhao, L.; Nasajpour, M. Secure Smart Communication Efficiency in Federated Learning: Achievements and Challenges. *Appl. Sci.* **2022**, *12*, 8980. https://doi.org/10.3390/app12188980

Academic Editor: Corrado Santoro

Received: 14 July 2022
Accepted: 2 September 2022
Published: 7 September 2022

1. Introduction

A machine learning (ML) model for a specific task is created utilizing the unprecedented amount of data that are generated from devices. These data can be used in various tasks, such as achieving process optimization [1], gaining insight discovery [2], and decision making [3]. Some examples of devices that generate data can be smartphones and wearable devices, smart homes, etc. Traditionally, to implement ML predictive models, the data would need to be transferred to a central server where it could be stored and used for training and testing ML models designed for specific tasks [4]. However, the imposition of privacy and data-sharing policies, such as global data protection regulations (GDPR), caused various restrictions on using data centrally [5]. As a result, the traditional centralized method of transferring data to the server could present more challenges. In addition to this, other computational issues are present with this traditional approach, and it is critical to deploy an alternative solution for creating reliable ML models within various tasks [4].

Federated learning (FL) is an innovative way to implement ML algorithms in a decentralized setting, and was introduced by a research team at Google for the first time [6]. FL attempts to answer the question: Can machine learning models be trained without needing to transfer user data onto a central server? [7]. FL aims to propose a more collaborative

approach to ML while preserving user privacy by training the data in a decentralized environment rather than a central server [8]. This collaborative learning method can be explained by using data generated from different participants, such as hospitals, research centers, or even mobile end-users, where the participants actively collaborate in the training procedure. For example, different hospitals generate data from their smart devices and/or equipment. Using these data can be utilized to create an ML model for a specific task. However, due to data-sharing policies, such as the health insurance portability and accountability act (HIPAA) and GDPR, it would not be possible to access all data in one place. A single hospital may not also generate enough data for ML training tasks, possibly leading to a biased or unreliable ML model [9]. This is where FL and its collaborative nature can thrive. Each hospital can train its own independent ML model that gets uploaded to a central server where all models average out as a global ML model for the assigned task. Figure 1 demonstrates an application of FL for a healthcare system where hospitals participate in training a global model as discussed above.

In general, computational power, training time, and, most importantly, security and privacy are different challenges that the traditional centralized-based ML approaches deal with. FL provides an effective way to preserve user security and privacy concerns by decentralizing data from the central server to end-devices, enabling AI benefits to all FL participants. In fact, rather than relying on the centralized server for training the ML model, in FL, the training process takes place on end-devices directly. Although we still have the concept of a central server in FL, the training procedure is mainly carried out by end-devices locally in order to enhance security and privacy concerns associated with transferring data. In addition, in an FL environment, the computational power is also split amongst parties within the federated network using the decentralized algorithms. FL is considered as one of the growing ML fields in recent years, where its security and privacy features are perfectly aligned with user data protection laws [10]. These security and privacy-preserving promises of FL technology have attracted much attention from different data-sensitive domains, such as healthcare [11], where the sensitive data are stored locally but, at the same time, the data are used in the training process.

Since FL was introduced, there has been a growing interest in the research communities to explore the opportunities and capabilities of FL in different domains. A range of industries are researching various examples of the FL capability to maximize the benefits of this decentralized approach, including transportation [12,13], autonomous vehicles [14], and internet of things (IoT) applications [15–18]. Gboard by Google for predictive texting is another simple and reliable implementation of FL, where a federated averaging algorithm is utilized to train the ML model over the on-device decentralized data to improve predictive texting results while also reducing the privacy and security risks of the sensitive user data compared to centralized ML approaches [19–22].

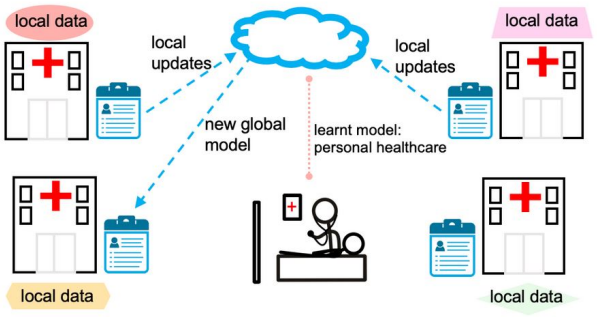

Figure 1. How FL can be used to improve predictive healthcare ML using sensitive data across multiple hospitals [23].

However, with the introduction of any new technology, the more benefits reaped from its capabilities, the more challenges that are exposed, and FL is not an exception. There has been a significant amount of challenges regarding the security and privacy of

FL. When it comes to security, there are three main concepts that should be considered in designing an FL environment, including confidentiality, integrity, and availability. Authors in [24] demonstrated the current vulnerabilities to the FL environment as communication protocols, data manipulation, central server, aggregation strategy, and implementation team. Due to such vulnerabilities, FL could be faced with various types of threats, including poisoning, inference, backdoor attacks, etc. Some of the these threats are shared with traditional ML models, whereas some of them are dedicated to FL settings. There have been many efforts to enhance the security and privacy preservation across different settings for FL, including secure multi-party computation [25], differential privacy [26], adversarial training [27], etc., and they are still in progress. The readers are encouraged to visit the referred surveys to obtain a greater understanding of the security and privacy aspects of FL.

As FL takes place in a distributed environment, end-devices are required to be connected over a reliable network to constantly share their updates. Factors such as end-device network connections operate at substantially lower rates when compared to network connections that are available at a data center. This form of unreliability in FL communication could be potentially expensive in the training process [28]. Therefore, some research has been conducted towards making the communication factor of an FL environment more efficient. For example, quantization and sparsification are two model compression techniques that can be integrated with the FL averaging algorithms to make communication more effective. It is an interesting area of research and there have been various efforts to make FL communication more reliable, but very limited surveys are found on this topic. Therefore, this survey paper aims to present a summary of recent research conducted on the subject. In this paper, the current challenges, proposed models, and possible methods for enhancing the FL models' efficiency with respect to communication are presented and discussed.

The remainder of this paper is structured as follows. Section 2 provides an introduction to the FL system and how it functions. Section 3 introduces and answers the research questions (RQs). Sections 4 and 5 review the key findings in this paper, where it navigates towards sharing future expectations, concentrating on this topic of research.

2. Problem Statement

2.1. Background

Federated learning is a learning paradigm that was introduced by Google in 2016 and has gained much attention in the last couple of years in different domains [6]. FL can be considered as a natural extension of conventional ML, where it addresses data governance and data access challenges. FL is a kind of local training at the different clients or participants at different institutions/devices that receive an initial model from a global server and then collaborate in the learning task using their local datasets. In fact, in FL, each participant actively trains and fine-tunes their model on its local dataset and shares the local updates that will be aggregated in the global model. In this procedure, the local steps that are so important for the ML training remain the same as conventional ML; however, the participants collaborate on the global model and everybody gains insights from others without communicating directly.

In this subsection, a general overview and the main components/entities available in an FL environment are introduced. Additionally, the FL processes from a communication perspective are presented.

2.2. The Components of FL Systems

The FL system is a learning process where users/participants can collaboratively train the ML model [29]. As described by [30], there are two main entities or components that are present in an FL environment: (i) the data owners or participants and (ii) the central server. The data owners produce the data on their end-devices (e.g., mobile phones), and these data generated by the owners are kept private to them and do not leave the device. Therefore, each participant or device has their private dataset. The central server is where

the global model stays. It is where the original model is stored and shared with all of the participants that are connected across the FL environment. This model is then a local model for each device, i.e., is trained on that independent device's dataset. Once all of the device training is complete and model parameters are obtained, each device uploads its local model back to the central server. Here, at the central server, the local updates are aggregated by applying aggregation algorithms that take place in order to generate a new global model [31–33].

2.2.1. The Processes of an FL System

FL allows for a promising approach towards collaborative machine learning while preserving privacy [34]. The process of an FL establishes communication between the two main entities, including participants and the central server. Participants communicate with the central server by downloading the local model, the light version of the global model, from the central server [35]. Each participant trains the local model based on its local dataset and improves it. Each trained model is then uploaded to the central server, where aggregation of all local trained models occurs for the next round [34]. As discussed above, a simplified overview of the FL system is depicted in Figure 2.

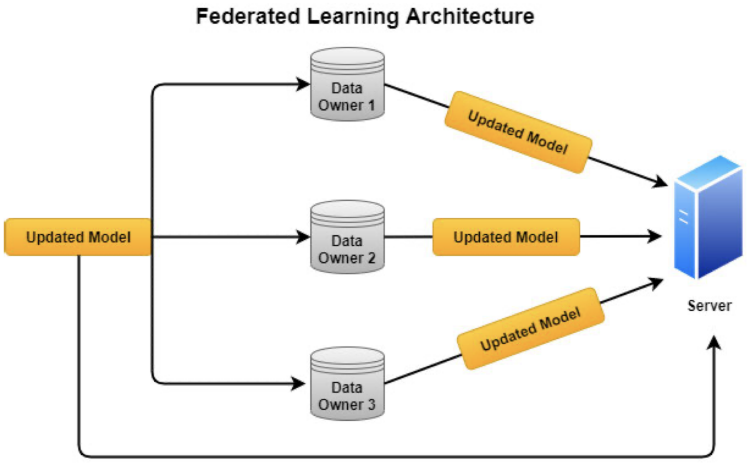

Figure 2. A general overview of how FL works [8].

2.2.2. The Different Types of FL Systems

The base concept of FL is to be able to utilize data that are shared across multiple devices. There are a few ways that these data can be distributed across devices, such as being partitioned by examples or partitioned by features [28]. The categorization of these data can be a prerequisite step for building an FL environment [24]. Some characteristics of data distribution are factors such as heterogeneous data and the clients' participation. In this subsection, a brief introduction is presented about the few FL settings that can be applied based on the distribution of data and other characteristics.

There are a few types of FL systems based on the way the data are distributed across the environment [36]. They can be categorized as:

- *Horizontal Federated Learning:* This type of FL system is when the data from various devices have a similar set of features in terms of the domain but with different instances. This is the original sense of FL learning, where data from each domain are homogeneous, as depicted in Figure 3. Then, domains contribute together to train the global ML model [24]. This can be explained using the original example that is presented by Google [6], wherein the global model is an aggregate of multiple locally trained participating devices [37].

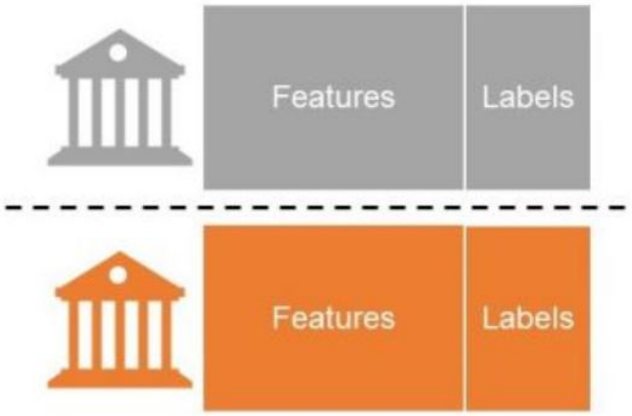

Figure 3. Horizontal FL [37].

- *Vertical Federated Learning:* The data distributed across in a vertical FL setting are common data between unrelated domains. This could perhaps be called a feature-based learning scheme as the datasets involved in the training process perhaps share the same sample ID space but may differ in feature space. An example could be where a bank and an e-commerce business in the same city have a mutual user base, as shown in Figure 4. The bank has user-sensitive data, such as credit card ratings or revenue. At the same time, the e-commerce business has a purchase and browsing history. Here, two different domains can use their data to maybe create a prediction model based on the user and product information [29].

Figure 4. Vertical FL [37].

- *Federated Transfer Learning:* This type of system is different from the aforementioned systems, where neither the samples nor the features have many similarities [38]. An example could be where two data sources, such as a bank in the United States and an e-commerce business in Canada, are restricted by geography but still have a small intersection with each other, being different institutions, similar to a vertical FL. However, this is just the method of partitioning the data by the ML model being similar to the traditional ML method of transfer learning, where the ML model used is a pre-trained model on a similar dataset. This method can provide better results in some cases compared to a newly built ML model [24]; this is further shown in Figure 5.

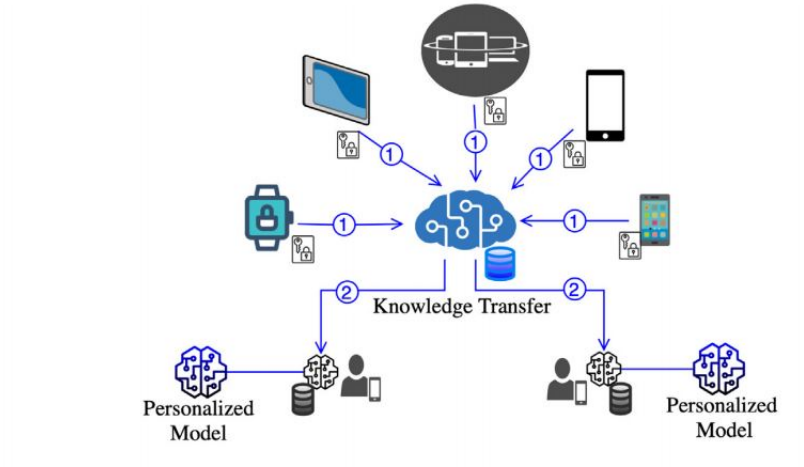

Figure 5. Federated transfer learning [24].

2.3. Publication Analysis

Over recent years, federated learning has become a driving research topic for many researchers, especially those interested in decentralized machine learning techniques and adhering to the privacy policies that have been imposed. The research catalog can often be presented in the form of survey papers. In this subsection, multiple recently published and conducted works within this domain are presented. Additionally, the impact of this paper compared to other works is highlighted.

Authors in [8] provide a comprehensive and detailed survey on FL. This research conducted by authors is quite thorough, with an introduction of FL and the different types of FL systems that exist. The authors provide a study with a focus on the software that allows for FL to happen, the hardware that it platforms, the protocol, and the possible applications for the emerging technology.

Similarly, the authors in [39] introduce the implications and the practicality of FL in healthcare, providing general solutions. The authors discuss and introduce some open questions towards data quality, model precision, integrating expert knowledge, etc. A survey that presents some challenges that can be present in the integration of FL is presented in [30]. The authors discuss mobile edge computing (MEC), FL's implementation, and how it could act as a dependable technology for the collaborative machine learning approach over edge networks.

In another effort, Yin et al. [40] focus on privacy aspects of FL and propose a hybrid approach to guarantee participants' privacy. They provide more balanced privacy-preserving for different data sets by presenting a Bayesian differential mechanism in FL. In the proposed model, a new encryption method is introduced to ensure that the central server cannot obtain the gradient parameters of each participant. Additionally, they utilized a sparse differential gradient to enhance both data transmission and storage efficiency.

Another work, conducted by authors in [41], presents an overview of FL along with its categorization based on data-partitioning techniques. The authors demonstrate the recent improvements and challenges within the most common FL algorithm, called FedAVG. Some of these challenges include massive communication costs, data heterogeneity, etc. They provide a review of the current possible optimization techniques concerning three main challenges in FL, including communication costs and statistical and structural heterogeneity [41]. Moreover, they discuss privacy in two aspects: current risks or attacks on FL and technologies to prevent such privacy challenges. Lastly, the authors reviewed the state-of-the-art applications of FL in the three main domains of mobile devices, industrial engineering, and healthcare [41]. Some instances could be detecting the chance of

hospitalization in a patient dealing with heart disease or classifying the stage of diabetic retinopathy among patients using a federated transfer learning approach [42,43].

Li et al. [44] conducted a review on the primary FL systems. This begins with an introduction to FL and its components. Moreover, they discuss the details of FL with six main perspectives, including data partitions, ML models, privacy, communication, FL types, and the purpose of developing FL. This paper also demonstrates the current open-source FL systems, along with their specifications. Lastly, the authors briefly describe FL's possible applications within domains of mobile service, healthcare, and finance [44].

In another work carried out by Pfitzner et al. [45], authors focus on the applications of FL for preserving data privacy within the medical field [45]. Authors conduct their survey based on a structured method for collecting and utilizing the papers in their review. Along with their literature review of the FL approaches, they discuss conducted research studies on FL's main concepts, similar to the above approaches [45]. Additionally, they demonstrate current attack approaches and defense techniques for preserving the FL model. Authors discuss their work within a digital health perspective by describing limitations when adopting FL in healthcare [45].

The authors in [46] conducted a survey on federated learning and the potential threats that might be available in this approach. As with any discovery of research and advancement in technology, there are always vulnerabilities and potential security threats that can lack robustness. The survey paper [46] introduces the risk of privacy and how that could potentially be exploited in an FL setting. Furthermore, the paper presents a brief review on how poisoning attacks can interfere and manipulate the outcome of an FL training model, such as data poisoning and model poisoning. Similarly, inference attacks can also lead to privacy leakage. Furthering this research topic, another survey expands on the threats towards FL as a survey by creating research questions regarding security and privacy and answering them by providing a comprehensive list of threats and attacks [24]. The authors also bring to the surface some unique security threats that exist in the FL environment and what defensive techniques can be implemented towards such vulnerabilities [24]. Defense techniques, such as proactive defenses, are a way to guess the threats and the potential risks associated with them. The authors present SNIPER, knowledge distillation, anomaly detection, and moving target defense solutions.

As discussed in recent review papers conducted on FL, many include similar categorizations to the ones above. In another research conducted by authors in [47], FL is discussed in three main aspects, including the design, challenges, and applications. Apart from all of the above details reviewed for FL, the authors review the works carried out using personalization techniques to achieve better models. They describe strategies and techniques to reduce the FL models' communication costs [47].

More surveys are conducted on the topic [44,48], and even though they introduce the topic of communication and the challenges that may be present, there has not been a dedicated research or survey paper written towards the challenges present in communication and the ways to make it efficient. This paper aims to bridge that gap by solely focusing on communication.

A list of recent surveys over the past few years is mentioned in Table 1. There is a common theme with most survey papers: introducing the technology, presenting the applications, and addressing the security and privacy benefits and concerns. In most papers, the topic of communication is introduced too, though it is only a brief surface-level part of the paper. This survey paper aims to be the bridge in that gap and presents a survey paper that focuses solely on the communication component for FL.

Table 1. Publication analysis of survey papers over the past couple of years.

Reference	Year	Objective	Security	Privacy	Communication	Challenges	Future
[29]	2019	Providing a survey of existing works on FL, discussing frameworks, concepts, and applications.	✓	✓	✗	✓	✓
[46]	2020	Introduction of concept of FL, predominantly covering threat model attacks.	✓	✓	✗	✓	✓
[8]	2020	An FL survey focusing on hardware, software and technologies, and real-life applications	✓	✓	✓	✓	✓
[30]	2020	Introduction of FL, presenting some existing challenges and their solutions	✓	✓	✓	✓	✓
[24]	2020	A survey on security and privacy of federated learning	✓	✓	✗	✓	✓
[48]	2020	A detailed survey introducing FL and the challenges.	✓	✓	✓	✓	✓
[41]	2020	An overview of FL characteristics and applications within three specific domains	✓	✓	✓	✓	✓
[39]	2021	A survey of FL in healthcare, covering common topics of introduction of technology, challenges, etc.	✓	✓	✓	✓	✓
[24]	2021	A comprehensive survey posing research questions with regard to FL and privacy and security	✓	✓	✓	✓	✓
[44]	2021	A thorough categorization of FL based on six main aspects to enable effective deployment of FL models	✓	✓	✓	✓	✓
[45]	2021	A systematic literature review of FL research studies with a concentration on the medical domain	✓	✓	✓	✓	✓
[47]	2021	A comprehensive review of three main aspects of FL: design, applications, and challenges	✓	✓	✓	✓	✓

3. Research Questions and Communication Efficient Methods

Communication between the participating devices and the central server is essential for an FL environment. The model is communicated over rounds of communication that download, upload, and train the ML model. However, as mentioned, this comes with its challenges. To gain more insight into the challenges present in an FL environment regarding communication, two research questions are proposed. The first **RQ1** aims to provide a brief analysis of *What are some of the challenges that are presented in FL with regard to communication?* The second **RQ2** provides an in-depth analysis of various methods and approaches that can be implemented to answer *How can communication be more efficient in an FL environment?*

To thoroughly conduct our survey based on the defined research questions, we explored various libraries, including Google Scholar, Springer Link, Science Direct, ACM Digital Library, IEEE Xplore, and ArXiv, using specific keywords such as federated learning, communication, etc. Consequently, this paper presents a comprehensive survey of the current challenges and its applicable solutions within FL environments.

3.1. RQ1-What Are Some of the Challenges Presented in FL with Regards to Communication?

As mentioned in Section 2 regarding the general working of the FL environment, the central server shares the light version of the global model across the network to all of the participating devices. The total number of devices participating in this FL environment can sometimes be in the millions, and the bandwidth over which the devices are connected to the environment can be relatively slow or unstable [49]. In an FL training environment, there can be many rounds of communication between the central server and all of the participating devices. Over a singular communication round, the local model is shared across all of the devices in the FL environment. Each participating device downloads the local model to train it on their local dataset. A version of that is uploaded back to the environment to the central server. Therefore, there is a constant downlink and uplink during the communication rounds [50]. However, due to the limited bandwidth, energy, and power on the device end, these rounds of communication can be slow [48]. Even with these challenges, the overall communication cost of sharing model updates is relatively lower than sharing copious amounts of data from the devices to a central server; it is still important to preserve the communication bandwidth further to make it more efficient [24]. This subsection introduces the overheads [51] or challenges that could create a communication bottleneck in an FL environment.

- *Number of participating devices:* Having a high number of participating devices in an FL environment has its advantage, wherein the ML model could be trained on more data and there could be a possible increase in performance and accuracy. However, the large number of devices participating in multiple FL training rounds at the same time could create a communication bottleneck. In some cases, a high number of clients could also increase the overall computational cost [24,52].
- *Network bandwidth:* In contrast to the traditional ML approach, the FL approach reduces the cost substantially; however, the communication bandwidth still needs to be preserved [24]. The participating devices may not always have the bandwidth needed. They could be participating under unreliable network conditions. Factors such as having a difference between the upload speed and download speed could result in delays, such as model uploads by participants [30] to the central server, which could lead to a potential bottleneck, leading to disrupting the FL environment.
- *Limited edge node computation:* The computation is now dependent on edge devices rather than powerful GPUs and CPUs. The edge devices could have limitations towards computation, power resources, storage, and limited link bandwidth [53]. The authors in [53] compared the training time between a central server and an edge device. They elaborated that an image classification model with over 60 million parameters can be trained in just a few minutes over a GPU, reaching speeds of 56 Gbps. However, even with a powerful smartphone connected over 5G, it could take much longer, reaching an average speed of 50 Mbps.
- *Statistical heterogeneity:* Another possible source for a communication bottleneck or where communication costs can rise could be statistical heterogeneity, where the data are non-independent and identically (non-i.i.d.) distributed [54]. In an FL environment, the data are only locally present on each participating device. They are gathered and collected by the participants on their independent devices based on their usage pattern and local environment. An individual participant's dataset in an FL environment could not be representative of the population distribution of the other participants and their datasets [6]. The size of data gathered and distributed amongst devices can typically vary heavily [55]. Therefore, this type of fluctuation in the size of the dataset could affect communication by causing a delay in model updates and other attributes. A device with a larger dataset could take longer to update, whereas a smaller one could be carried out with updates. However, the global model might not be aggregated until all individual client models are trained and uploaded, causing a bottleneck.

The aforementioned constraints listed are just some of the discovered challenges that could be possible sources towards creating a communication bottleneck.

3.2. RQ2-How Can Communication Be More Efficient in an FL Environment?

In order to train the global ML model with decentralized data, the global model needs to be downloaded on the participating devices on that federated network. This allows the data generated by those remote devices to remain preserved on the device while subsequently improving the ML model. The steps that make this possible can be summarized into three communication steps: (1) the global model needs to be shared across devices within the federated network, which can sometimes be millions of IoT devices of mobiles phone; (2) the model is then downloaded by the devices and trained locally on-device on the private dataset that is generated by those devices; (3) the ML model is uploaded back to the central server, where it is pooled with numerous other models that have been uploaded to aggregate them all together and find a federated average to generate a new and updated global model. Figure 6 depicts the whole process of uploading and downloading, along with the aggregation of local models. Considering the steps of communication that take part in obtaining an improvement in the global ML model, it is important to seek out the most communication-efficient methods that could make the transfer of data from (1) to (2) and on to (3).

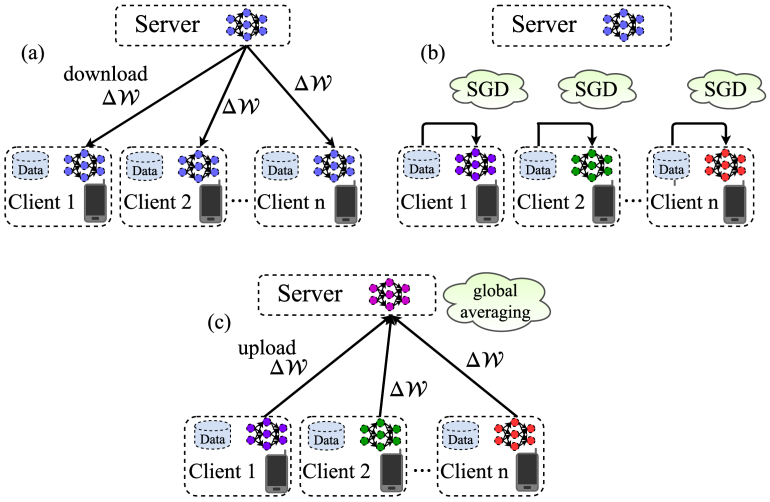

Figure 6. Ref. [55] illustrates a complete round of distributed stochastic gradient descent (SGD) model. In (**a**), clients on the federated network synchronize with the server and download the global ML model. In (**b**), the global model is trained by each client on their local data, adjusting the weighted average of the ML model as per. In (**c**), the new average achieved by each client is updated onto the server, where each update is used to obtain a new weighted average for the global model.

In this subsection, the recent findings and conducted works on improving the communication efficiency in ML models over the federated network are presented: findings that are tackling constraints where devices can drop out due to a poor or limited network bandwidth, along with other constraints where data are sampled differently and how these could affect communication. Then, some of the main research methods towards enhancing the communication efficiency are discussed, such as local updating, client selection, reduction in model updates, decentralized training and peer-to-peer learning, and various compression schemes.

3.2.1. Local Updating

Distributed ML that exists in data centers has become popular when integrating mini-batch optimization methods. However, there are still some constraints and limitations on flexibility. This form of batch optimization has constraints over a more federated setting that includes both communication and computation challenges [48,56]. Local updating in FL attempts to train ML models on each device with their own data instead of sharing data captured from devices with the ML model to be trained together. Consequently, each device is trained locally and sends its own updates to the decentralized server. Then, the updates will be aggregated in order to create the global model.

However, the overall objective of local updating is mostly focused on the aforementioned point (2), i.e., fitting and training the ML model locally on-device using the data generated by those devices. However, computing locally over a singular communication round and then applying those updates to the central server is not that efficient due to unexpected dropouts of devices or synchronization latency due to poor network connectivity [57]. Authors in [6] also highlight that it is very important to have a form of flexibility when considering the optimization method: flexibility towards client participation and local updates. There are a few ways to achieve this flexibility and, in turn, improve the overall communication efficiency. Techniques such as primal-dual methods can offer a form of flexibility, where local devices can use local parameters to train the global model locally to find arbitrary approximations [58]. This leveraging and breaking down of the overall objective of the global model can allow for problems of training to be solved in parallel over each communicated round [56]. In this subsection, some methods towards communication efficiency that have come to the surface regarding local updating techniques are reviewed.

In an FL setting, the data can be distributed unevenly across devices and may not always be identical. This can be considered as due to the data across the FL environment being distributed in a non-independent and identically distributed manner, i.e., non-iid. Having the datasets in a non-iid state could challenge the testing accuracy of an FL model [57]. The testing accuracy of a locally trained ML model is important as it will contribute to the global model. Therefore, having an accuracy of a local model that performs poorly could also deteriorate the overall performance of the global model.

In [59], the authors introduce their way of combining FL with hierarchical clustering (HC) techniques to reduce the overall communication rounds within the FL protocol. The proposed clustering step can cluster and separate the clients by the similarity of the local updates to the global model. These similarities are the weights that are achieved by locally updating the local model on each device. Upon testing and comparing their integrated FL and HC technique, the authors concluded that the communication rounds are reduced as per a comparison on the Manhattan distance metric [59,60].

Due to the large number of connected devices that attempt to communicate their locally updated model parameters with the central server—step (2)—this could cause a communication bottleneck due to bandwidth in some instances [61]. The authors in [61] introduce a federated periodic averaging and quantization framework (FedPAQ) as a new way to counter this problem. In the proposed framework, the periodic averaging feature is introduced, where it helps to reduce the number of communication rounds. In contrast to other training methods, where each participating device sends their ML models to synchronize through the parameter servers over each iteration, resulting in increasing the number of communication rounds between the devices and central sever, the parameters and local updates of each device can be synchronized with the central server, where a periodic average of models takes place. This is achieved by adjusting the parameter that corresponds to the number of iterations that occur locally on the device itself. Other features, such as partial node participation and a quantized message passing of the FedPAQ, also reduce communication overhead. The quantized message passing is further discussed in the compression schemes subsection.

In FL, the FedAVG algorithm is used for its simplicity and, in turn, is about to reduce communication costs in an FL environment. The algorithm tackles the challenges presented in communication by performing numerous updates on available devices before communicating with the central server. However, in some cases, as the authors in [62] state, over heterogeneous data, the FedAVG algorithm could introduce 'drift' towards the updates of each client or device that is participating. This 'client drift' will result in a slow and unstable convergence and could persist if all clients participate in training rounds with full batch gradients. The authors ran their analysis and determined the FedAVG algorithm full batch gradients and matching lower bounds over no client sampling, concluding that it is slower than the stochastic gradient descent over some parameters [62]. To overcome this challenge, the authors created a framework called a stochastic controlled averaging (SCAFFOLD) algorithm that can fix the drift [62]. The SCAFFOLD algorithm performs well for their analysis and provides reliable convergence rates as SGD, even for non-iid data. It also takes advantage of the similarity that exists within the clients, and reduces communication overload [63,64]

An approach introduced by authors in [65] called FedDANE tackles some of the practical constraints that are present with FL. The approach is a culmination of methods introduced in [63,64]. FedDANE collects gradient updates from a subset of devices during each communication round. FedDANE also works well with low client participation settings.

In summary, local updating is an essential part of federated learning. Each device needs to compute local updates so a better overall global model can be generated and all participating devices can benefit from it. Making the process of local updating more efficient could indeed make communication more efficient.

3.2.2. Client Selection

Client selection is an approach that plays a key role for FL environments. This technique determines which devices should be trained and which parameter could be specified to be aggregated for the local updates. It can be implemented to make communication more efficient by reducing the costs and restricting the number of participating devices so that only a fraction of the parameters are updated over the communication round [39]. In this subsection, conducted works on the client selection and its benefits towards improving the communication overload are presented.

An approach introduced by the authors in [66] tackles the limitations that are present in communication over an FL setting. They create a multi-criteria client selection model, called FedMCCS, for IoT devices over an FL setting. The FedMCCS approach considers all of the specifications and network conditions of the IoT device for client selection. Factors such as the CPU, memory, energy, and time are all considered for the client resources to determine whether the client would be able to participate in the FL task. The FedMCCS approach considers more than one client and, over each round, the number of clients participating is increased. From their analysis, FedMCCS, compared to other approaches, can outperform by reducing the total number of communication rounds in order to achieve a reasonable accuracy.

Another factor that is a vital property of FL, according to the authors in [67], is the varying significance of learning rounds. This comes to the surface when the authors realize that the learning rounds are temporally interdependent but have varying significance when achieving the final learning outcome. This conclusion comes from running numerous data-driven experiments; the authors create an algorithm that utilizes the wireless channel information but can achieve a long-term performance guarantee; the algorithm provides a desired client selection pattern adapted to network environments.

The authors in [68] introduced a framework that specifically tackles the challenges when it comes to client selection. The authors called their framework FedCS. Their goal is that, when it comes to client selection in a standard FL setting, it can sometimes be a random selection of clients (or devices). However, with their approach of client selection, they break it into two steps. First, a resource request, where client information, such as the state of the wireless channel and computational capacities, etc., are requested and shared with the central server. Second, in order to estimate the time required, distribution and scheduled update and upload steps are taken. With their framework, the overall approach of the client selection is to allow the server to aggregate as many clients within a certain time frame, knowing that factors, such as data, energy, and computational resources, i.e., used by the devices, could better meet the requirements of a training task and possibly affect the energy consumption and bandwidth cost.

To better assess whether a client can sufficiently participate or not, a resource management algorithm introduced by the authors in [69] adopts a deep-Q learning algorithm that can allow the servers to learn and make optimal decisions without having to know prior network knowledge. Their mobile crowd machine learning (MCML) algorithm addresses the constraints present in mobile devices, reducing the energy consumed and making the training and communication time more efficient.

Authors run their analysis, providing a method that practices biased client selection strategies [70]. As per their analysis, the selection of bias can affect the convergence speed; the bias of their client selection works towards clients that have a higher local loss and achieve a faster error convergence. With this knowledge, the authors create a communication and computation-efficient framework called power-of-choice. Their framework has the agility to trade off between the bias and convergence speed. The results that the authors achieve are computed relatively faster and provide a higher accuracy of results than the baseline random selection methods [70].

Uplink and downlink communication with participating clients in a federated network is necessary. Having to communicate with clients that have a dependable network bandwidth and energy sources could aid in achieving a well-trained global model more

efficiently. Implemented client selection techniques can aid in reducing the cost of the overall communication that is needed to achieve a dependable global model.

Considering mobility participants in FL, load-balancing is a key point when managing available resources and minimizing the training and communication delay. Authors in [71] proposed a load balancing and computation offloading technique for mobile-edge computing systems that can also be implemented in the FL environment.

3.2.3. Reducing Model Updates

Once the global model is downloaded by connected devices in the FL environment, each device starts to train the devices locally. As each device computes and updates the model, these updates are communicated back to the central server [6]. The number of communication rounds between the devices and central server can be costly, and perhaps having fewer but more efficient model updates could be a solution. This section introduces some techniques that discuss a possible reduction in communication for these model updates and could potentially reduce the cost.

The authors in [72] introduce an efficient way of training models; their proposed approach adapts to the concept of drifts and trains models equally well through different phases of model training. Their approach leads to reducing the communication substantially without depreciating the model's predictive performance.

In another study [73], the authors introduce a partitioned variational inference (PVI) for probabilistic models that work well over federated data. They train a Bayesian neural network (BNN) over an FL environment that is allowed for both synchronous or asynchronous model updates across many machines. Their proposed approach, and the integration of other methods, could allow for a more communication-efficient training of BNN on non-iid federated data.

In contrast to most of the current FL methods that include iterative optimization techniques over numerous communication rounds [6,74], the authors in [75] introduce a one-shot communication round approach where only a single round of communication is conducted between the central server and the number of connected devices [75]. Instead of computing increments, the authors suggest that each device trains a local model to completion and then applies ensemble methods to capture information regarding device-specific models effectively. Applying ensemble learning techniques could be better suited for global modeling than averaging techniques [75].

In an FL setting, the rate at which model convergence occurs can sometimes take a large number of communication rounds, creating a delay towards model training while simultaneously increasing network resources [76]. An intelligent algorithm called FOLB is introduced by the authors in [76]. The algorithm performs a smart sampling of participating devices over each round to optimize the expected convergence speed. The algorithm can estimate the participating device's capabilities; this is achieved by adapting to devices' aggregations.

If the mere frequency in which the model updates are shared is reduced, the overall communication between the devices and the central server is also reduced. Having an effective way of determining how these updates are computed reduces the overall communication rounds that are needed. Fewer rounds could result in a greater efficiency.

3.2.4. Decentralized Training and Peer-to-Peer Learning

FL, in a way, is like practicing ML in a decentralized manner. However, FL does allow for a more peer-to-peer learning approach wherein each trained node can benefit from the other node that is trained on the FL network. Even in decentralized training, similar challenges of communication exist, and, in order to tackle it, different methods such as compression [77] can be used. Herein, this subsection presents how decentralized or peer-to-peer learning is utilized or integrated into an FL environment.

In an FL environment, as aforementioned, there is a central server that has the original model. The central server is also where all of the devices that are connected to the network

update their model. Essentially, all devices that are participating in the FL environment are connected to the central server. As stated by [48], the star network is the predominant communication topology in an FL setting as shown in Figure 7. However, there is some research on a different communication topology, where the participating devices only communicate with one another, a peer-to-peer learning experience or decentralized training network, and whether this would be a more efficient way to communicate in an FL environment. The authors in [48] state that, in traditional data center environments, decentralized training can appear to be faster than centralized training, especially when constraints of a low bandwidth and high latency are faced when operating on networks. However, this is not to say that decentralized training does not have its constraints: the authors in [78] state that the computation time on nodes can slow down the convergence of a decentralized algorithm. In addition to this, sometimes, a large communication overhead could also further mitigate this. To overcome these constraints, the authors in [78] proposed the QuanTimed-DSGD algorithm, a decentralized and gradient-based optimization that imposes iteration deadlines for nodes, where nodes exchange their quantized version of the models.

In the FL environment, peer-to-peer learning could be used, where each device only communicates with its neighbours and updates. The participating devices or clients update their model on their dataset and aggregate it along with the model updates from their neighbours [39]. Furthering that, the authors in [79] build a framework for an FL environment towards a generic social network scenario. Their online push-sum (OPS) method handles the complex topology while simultaneously having optimal convergence rates.

Similarly, [80] proposes a distributed learning algorithm where there is no central server but, instead, the participating devices practice a peer-to-peer learning algorithm to iterate and aggregate model information with their neighbor to estimate the global model parameters collaboratively. The authors put forward an assumption, suggesting that the FL setting wherein a central server exists and communicates with the global model can incur high communication costs. In their approach, the devices are already distributed over the network, where communication occurs only with their one-hop neighbors.

The authors in [81] further provide another avenue for peer-to-peer learning that does not depend on the central server as a single trusting authority. The authors [81] create a framework, BrainTorrent, that does not rely on a central server for training; their proposed peer-to-peer framework is designed to motivate a collaborative environment. According to their analysis, the absence of a central server makes the environment resistant to failure but also precludes the need for a governing central body that every participant trusts.

Theoretically, the application of decentralized training in an FL setting could reduce the communication cost when compared with a central server [48], though there is more research conducted on this and there are further avenues that it can take too.

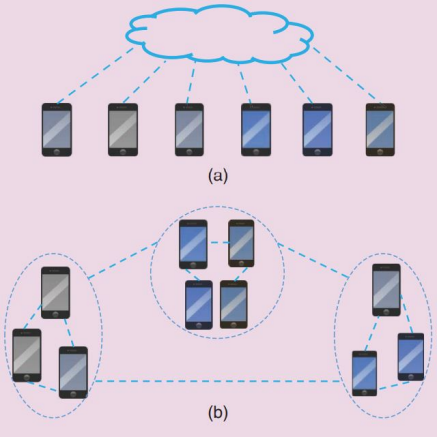

Figure 7. An overview of star-network topology. (**a**) star network, (**b**) decentralized network [48].

3.2.5. Compression Schemes

In addition to the local updating methods, some ML compression techniques may also be implemented to reduce the total number of rounds between the centralized server and the devices connected to the federated network. Compression schemes such as quantization and sparsification that are sometimes integrated with the aggregation algorithms can sometimes provide reasonable training accuracy results while, at the same time, also reducing the overall communication cost. The authors in [28] list the objectives set for compression schemes and federated learning, such as:

a. Reducing the size of the object / ML model from the clients to the server, i.e., that used to update the overall global model.
b. Reducing the size of the global model that is shared with the clients on the network, i.e., the model on which the clients start local training using the available data.
c. Any changes that are made to the overall training algorithm that make training the global training model more computationally efficient.

From the list of objectives, **A** could have the highest effect on the overall running time; therefore, reducing it would directly result in reducing the overall communication cost. The clients on a federated network generally have a slower upload bandwidth compared to the download bandwidth. Therefore, compressing the ML models and potentially reducing the uplink/downlink exchanges could result in reducing the communication cost [39].

In this subsection, compression schemes towards achieving communication efficiency in a federated environment are presented. First, sharing the research conducted towards compression schemes is discussed. Then, conducted works on concerned methods of sparsification and quantization are covered.

***Sparsification*:** Sparsification is a process of improving the model by removing unnecessary information, mainly when it comes to deep learning approaches. It aims to provide a faster and more efficient model that could enhance communications within specific techniques in ML and FL. This technique is a type of communication technology that can be implemented in an FL setting to compress the model when communicated across the server. Herein, the conducted research towards making integrating sparsification in an FL environment is discussed.

The authors in [55] present a sparse ternary compression (STC) method that not only adheres to the compression requirements of an FL setting and environment but also provides compression for both upstream and downstream communications. They run an analysis of FL models over various datasets and architectures. In their analysis, they conclude that some factors of an FL environment are hugely dependent on the convergence rate of the averaging algorithm. Moreover, the authors deduce that training on non-iid small portions of data or when only a subset of clients participate in communication rounds can reduce the convergence rate. However, the proposed model of STC is a protocol that compresses communication via sparsification, ternarization, error accumulation, and the optimal Golomb encoding. The robust technique provided by the authors in [55] converges relatively faster when compared to other averaging algorithms, such as FedAVG, over both factors of non-iid data and a lower number of iterations that are communicated. The proposed method is also highly effective when the communication bandwidth is constrained.

Sparse client participation is another challenge that needs to be overcome in an FL environment; authors have introduced a FetchSGD algorithm that can help towards achieving this [82]. FetchSGD overall aids with the communication constraints of an FL environment by compressing the gradient that is based on the client's local data. The data structure count sketch [83] is used to compress the gradient before it is uploaded to the central server. The count sketch is also used for error accumulation. According to the authors in [84], one key problem with regard to federated setting and communication is the communication overhead that is involved in parameter synchronization. The authors inform that this overhead wastes bandwidth and increases the training time while impacting the overall model accuracy. To tackle this, the authors propose a general gradient sparsification (GGS)

framework for adaptive optimizers. The framework consists of two important mechanisms: batch normalization updates with local gradients (BN-LG) and gradient correction. The authors determine that updating the batch normalization layer with local gradients could mitigate the impact of delayed gradients and not increase the communication overhead. The authors run their analysis over several models, such as AlexNet, DenseNet-121, CifarNet, etc., and achieve high accuracy results, concluding that gradient sparsification does have a significant impact in reducing the communication overhead. The authors in [85] introduce a compression scheme that is both a communication-efficient and deferentially private federated learning scheme. The authors call it CPFed. The challenge that the authors face when addressing both issues together is one that arises from data compression. Techniques used for compression can sometimes lead to an increased number of training iterations required for achieving some desired training loss due to the compression errors; however, differential privacy could deteriorate with regard to the training iterations. To overcome this paradigm, their proposed CPFed is based on a sparsified privacy masking technique that adds random noise to model updates along with an unbiased random sparsifier before updating the model. The authors can achieve a high communication efficiency through their proposed model.

The authors in [86] propose a compression technique that could drastically reduce the communication cost for a distributed training environment. The framework introduced by the authors is a sparse binary compression (SBC) technique; their method integrates techniques that are already present in communication delay and gradient sparsification with a novel binarization method. According to their research, the authors find that the current gradient information for training neural networks with SGD is redundant. Instead, the authors utilize communication delay methods introduced in [6] to introduce temporal sparsity, where gradients are not communicated after every local iteration. From their findings, the authors conclude that, in a distributed SGD setting, both the communication delay and gradient sparsity can be treated as independent types of sparsity techniques. These methods provide higher compression gains, though, in their findings, the authors also find a slight decrease in the accuracy.

Quantization: The communication between the devices and the central server includes sharing model updates and parameters that have occurred on the device end. This updating of parameters can be strenuous when it comes to up-linking the model. To aid with this, another compression method referred to as quantization can bring the model parameters to a reasonable size without compromising much on the model accuracy [87]. In this subsection, different techniques and frameworks that have integrated quantization towards achieving communication efficiency in the FL setting are introduced.

The authors in [61] utilize their framework, FedPAQ, to reduce the overall communication rounds and the bearing cost. A feature of their framework is to quantize message-passing. The communication bandwidth will mostly be limited in an FL setting, where a limited uplink bandwidth on the participating device end could increase the communication cost, making it more expensive. The authors in [61] employ quantization techniques [88] on the up-links; every local model is quantized before being uploaded, hence reducing the overall communication overhead.

Furthering and employing quantization techniques, the authors in [89] use them for their FL analysis. In contrast to typical FL models, where the global model from a central server is downloaded by all devices and then subsequently updated and so and so forth, the authors from their analysis discovered that applying quantization techniques to the global model can help towards making communication more efficient. The lossy FL algorithm (LFL) created by the authors quantizes the global model before it broadcasts and shares it with the devices. The local updates that take place on the device are uploaded on the global central server and are also quantized. The FL environment is hugely dependent on the bandwidth, and the authors in [89], for their analysis, study how well the quantized compressed global model performs in order to provide an estimate for the new global model from the local updates of the devices. They compare their results with another analysis of a

lossless standard approach of FL and conclude that their LFL and the technique of quantizing global model updates provide a significant reduction in the communication load.

Traditionally, compression algorithms are trained for a setting where there is a high-speed network, such as data centers [90], but, in an FL setting, these algorithms might not be very effective directly. To further tackle the communication bottleneck that is created due to the network and the other aforementioned reasons, another quantization compression technique is proposed by [91]. To achieve a trade-off between communication efficiency and accuracy, the authors in [91] propose a hyper-sphere quantization (HSQ) framework. HSQ can reduce the communication cost per iteration, which is ideal for an FL environment. The framework utilizes vector quantization techniques that show an effective approach towards achieving gradient compression and simultaneously not compromising regarding convergence accuracy.

In another study, the authors in [92] suggest that the communication channel and transfer of model parameters between the users to the central server has a throughput that can be typically constrained. The authors, while conducting their research, encountered that alternative methods used to aid with this could provide a dominant distortion of results. This leads [92] to create a quantization method that is more efficient towards facilitating the model transfer in an FL setting. Utilizing quantization theory methods, the authors design quantizers that are suitable for distributed deep network training. Understanding the requirements that are needed for a quantization FL setting, the authors can propose an encoding–decoding strategy. Their proposed scheme shows potential for an FL setting, as it performs relatively well compared to previously proposed methods. Furthering their research towards quantization theories, the authors in their review approach it by identifying unique characteristics regarding the conveyed trained models over rate-constrained channels [93]. The authors propose a universal vector quantization technique for FL: a quantization technique that would be suitable for such settings, calling it UVeQFed. For their research, the authors demonstrate that combining universal vector quantization methods can yield a system where the compression of trained models induces only a minimum distortion. Analyzing the distortion further, the authors determine that the distortion is reduced substantially as the number of users grows.

The authors in [94] introduce a hierarchical quantized federated learning technique that can leverage a client-edge-cloud network hierarchy and quantized model updates. Their heir-local-QSGD algorithm performs partial edge aggregation and quantization on the model updates, which can result in improving the communication efficiency in an FL environment. A summary table of the works presented in this section is given in Table 2.

Table 2. Research conducted towards reducing the overall communication costs and overheads in an FL.

Ref.	Section	Model and Technology	Remarks
[59]	Local Updating	Hierarchical clustering technique	An FL+HC technique separating client clusters similarity of local updates
[61]		FedPAQ	Using periodic averaging to aggregate and achieve global model updates
[62]		SCAFFOLD algorithm	An algorithm that provides better convergence rates over non-iid data
[55]	Compression Schemes-Sparsification	STC method	Providing compression for both upstream and downstream communications
[82]		FetchSGD	Compresses the gradient based on client's local data
[84]		General gradient sparsification (GSS)	Batch normalization layer with local gradients mitigating the impact of delayed gradients and not increasing the communication overhead
[85]		CPFed	A sparsified masking model providing compression and differential privacy
[86]		Sparse binary compression (SBS)	Introducing temporal sparsity, where gradients are not communicated after every local iteration

Table 2. *Cont.*

Ref.	Section	Model and Technology	Remarks
[61]	Compression Schemes-Quantization	FedPAQ	Using quantization techniques based upon model accuracy
[89]		Lossy FL algorithm (LFL)	Quantizing models before broadcasting
[91]		Hyper-sphere quantization (HSQ) framework	Ability to reduce the cost of communication per iteration
[92]		UVeQFed	Algorithm convergence of model minimizes the loss function
[94]		Heir-Local-QSGD	Leveraging client–edge–cloud network hierarchy and quantized models updates
[81]	Decentralized Training or Peer-to-peer Learning	BrainTorrent	A peer-to-peer learning framework where models converge faster and reach good accuracy
[78]		QuanTimed-DSGD	decentralized gradient-based optimization imposing iteration deadlines for devices
[66]	Client Selection	FedMCCS	A multi-criteria client selection that considers IoT device specification and network condition
[67]		Resource allocation model	Optimizing learning performance in how clients are selected and how bandwidth is allocated
[68]		FedCS	The framework allows the server to aggregate as many clients as possible within a certain time-frame
[70]		Power-of-choice	A communication and computation-efficient client selection framework
[72]	Reduced Model Updates	A decentralized deep learning model	Ability to handle different phases of the model training well
[73]		A partitioned variational inference (PVI)	A Bayesian neural network over FL that is synchronous and asynchronous for model updates across machines
[75]		One-shot federated learning	A single round of communication performed between central server and connected devices
[76]		FOLB	Intelligent sampling of devices in each round of model training to optimize the convergence speed

4. Discussion

This paper provides an introduction to FL while retaining focus on the communication component of FL. Communication is quite an essential part of the federated learning environment. It is essentially how the global machine learning model is transmitted from the central server to all of the participating devices. Similarly, the model is then trained on the local data that are available on the devices and then uploaded back to the central server. This constant communication requires many downloads and uploads using a reliable network bandwidth. As this is not always the case, a limited bandwidth or poor client participation can create a communication lag in completing the FL training. Techniques used to make the rounds of communication more efficient are shared in this paper. For example, a wide range of compression scheme methods mentioned in this paper can help to reduce the communication overheads and reduce the costs to some factor. Even though the methods presented in this paper make the communication front more efficient, they are all implemented using various resources, such as different computational power and datasets. There are dependable ways for communication rounds to be addressed. However, due to the range of solutions presented, it could be concluded that a culmination of techniques implemented together could result in the most communication-robust solution for an FL setting.

5. Conclusions and Future Expectations

Communication plays an important role in an FL setting, and various techniques, such as local updating, client selection, reduction in model updates, decentralized training, peer-to-peer learning, and compression schemes, are proposed to enhance the communication efficiency. This survey paper aimed to provide a bridge between the gap that was present on the topic of communication in FL. The paper mainly focused on the communication challenges and constraints in an FL environment, such as the bandwidth and limited computation. The challenges were proposed in the form of two RQs that both introduced the problems and provided a list of solutions that have been applied towards making communication more efficient. It is, after all, how all of the training of ML models occurs. Despite its importance, there is still limited research conducted compared to other aspects, such as privacy, security, and resources. For future expectations, it is critical to compose the aforementioned techniques effectively in order to overcome communication costs. Using a combination of techniques could be a future direction for research studies. In this case, the trade-off between the model's performance and communication could be studied. Additionally, more research is needed to improve the communication efficiency across FL settings.

Author Contributions: S.P.: supervision, conceptualization, methodology, investigation, writing—review and editing. O.S.: conceptualization, methodology, software, data curation, validation, investigation, visualization, writing—original draft. R.M.P.: supervision, methodology, validation, writing—review and editing. Q.Z.S.: methodology, writing—review and editing. G.S.: methodology, validation, writing—review and editing. L.Z.: methodology, writing—review and editing. M.N.: writing—review and editing. All authors have read and agreed to the published version of the manuscript.

Funding: This work was supported by U.S. SunTrust Fellow in Cybersecurity/Information Security Research Funding Program under Grant ST22-04.

Institutional Review Board Statement: Not applicable.

Informed Consent Statement: Not applicable.

Data Availability Statement: Data available on request from the authors.

Conflicts of Interest: The authors have no conflicts of interest to declare for this manuscript.

References

1. Weichert, D.; Link, P.; Stoll, A.; Rüping, S.; Ihlenfeldt, S.; Wrobel, S. A review of machine learning for the optimization of production processes. *Int. J. Adv. Manuf. Technol.* **2019**, *104*, 1889–1902. [CrossRef]
2. Pazzani, M. Comprehensible knowledge discovery: Gaining insight from data. In Proceedings of the First Federal Data Mining Conference and Exposition, Citeseer, Newport Beach, CA, USA, 14–17 August 1997; pp. 73–82.
3. Meyer, G.; Adomavicius, G.; Johnson, P.E.; Elidrisi, M.; Rush, W.A.; Sperl-Hillen, J.M.; O'Connor, P.J. A machine learning approach to improving dynamic decision making. *Inf. Syst. Res.* **2014**, *25*, 239–263. [CrossRef]
4. L'heureux, A.; Grolinger, K.; Elyamany, H.F.; Capretz, M.A. Machine learning with big data: Challenges and approaches. *IEEE Access* **2017**, *5*, 7776–7797. [CrossRef]
5. Albrecht, J.P. How the GDPR will change the world. *Eur. Data Prot. L. Rev.* **2016**, *2*, 287. [CrossRef]
6. McMahan, B.; Moore, E.; Ramage, D.; Hampson, S.; y Arcas, B.A. Communication-efficient learning of deep networks from decentralized data. In Proceedings of the Artificial Intelligence and Statistics, PMLR, Fort Lauderdale, FL, USA, 9–11 May 2017; pp. 1273–1282.
7. Truex, S.; Baracaldo, N.; Anwar, A.; Steinke, T.; Ludwig, H.; Zhang, R.; Zhou, Y. A hybrid approach to privacy-preserving federated learning. In Proceedings of the 12th ACM Workshop on Artificial Intelligence and Security, Los Angeles, CA, USA, 7–11 November 2019; pp. 1–11.
8. Aledhari, M.; Razzak, R.; Parizi, R.M.; Saeed, F. Federated learning: A survey on enabling technologies, protocols, and applications. *IEEE Access* **2020**, *8*, 140699–140725. [CrossRef] [PubMed]
9. Yazdinejad, A.; Srivastava, G.; Parizi, R.M.; Dehghantanha, A.; Choo, K.K.R.; Aledhari, M. Decentralized Authentication of Distributed Patients in Hospital Networks Using Blockchain. *IEEE J. Biomed. Health Inform.* **2020**, *24*, 2146–2156. [CrossRef] [PubMed]

10. House, W. Consumer data privacy in a networked world: A framework for protecting privacy and promoting innovation in the global digital economy. *White House Washington DC* **2012**, *1*, 120.
11. Rieke, N.; Hancox, J.; Li, W.; Milletari, F.; Roth, H.R.; Albarqouni, S.; Bakas, S.; Galtier, M.N.; Landman, B.A.; Maier-Hein, K.; et al. The future of digital health with federated learning. *NPJ Digit. Med.* **2020**, *3*, 1–7. [CrossRef] [PubMed]
12. Liu, Y.; James, J.; Kang, J.; Niyato, D.; Zhang, S. Privacy-preserving Traffic Flow Prediction: A Federated Learning Approach. *IEEE Internet Things J.* **2020**, *7*, 7751–7763. [CrossRef]
13. Yazdinejad, A.; Parizi, R.M.; Dehghantanha, A.; Karimipour, H. Federated learning for drone authentication. *Ad. Hoc. Netw.* **2021**, *120*, 102574. [CrossRef]
14. Pokhrel, S.R.; Choi, J. Federated learning with blockchain for autonomous vehicles: Analysis and design challenges. *IEEE Trans. Commun.* **2020**, *68*, 4734–4746. [CrossRef]
15. Zhao, Y.; Zhao, J.; Jiang, L.; Tan, R.; Niyato, D. Mobile edge computing, blockchain and reputation-based crowdsourcing iot federated learning: A secure, decentralized and privacy-preserving system. *arXiv* **2019**, arXiv:1906.10893.
16. Mothukuri, V.; Khare, P.; Parizi, R.M.; Pouriyeh, S.; Dehghantanha, A.; Srivastava, G. Federated Learning-based Anomaly Detection for IoT Security Attacks. *IEEE Internet Things J.* **2021**, *9*, 2545–2554. [CrossRef]
17. Attota, D.C.; Mothukuri, V.; Parizi, R.M.; Pouriyeh, S. An Ensemble Multi-View Federated Learning Intrusion Detection for IoT. *IEEE Access* **2021**, *9*, 117734–117745. [CrossRef]
18. Saharkhizan, M.; Azmoodeh, A.; Dehghantanha, A.; Choo, K.K.R.; Parizi, R.M. An Ensemble of Deep Recurrent Neural Networks for Detecting IoT Cyber Attacks Using Network Traffic. *IEEE Internet Things J.* **2020**, *7*, 8852–8859. [CrossRef]
19. Hard, A.; Rao, K.; Mathews, R.; Ramaswamy, S.; Beaufays, F.; Augenstein, S.; Eichner, H.; Kiddon, C.; Ramage, D. Federated learning for mobile keyboard prediction. *arXiv* **2018**, arXiv:1811.03604.
20. Chen, M.; Mathews, R.; Ouyang, T.; Beaufays, F. Federated learning of out-of-vocabulary words. *arXiv* **2019**, arXiv:1903.10635.
21. Yang, T.; Andrew, G.; Eichner, H.; Sun, H.; Li, W.; Kong, N.; Ramage, D.; Beaufays, F. Applied federated learning: Improving google keyboard query suggestions. *arXiv* **2018**, arXiv:1812.02903.
22. Ramaswamy, S.; Mathews, R.; Rao, K.; Beaufays, F. Federated learning for emoji prediction in a mobile keyboard. *arXiv* **2019**, arXiv:1906.04329.
23. Tian Li. Federated Learning: Challenges, Methods and Future Directions. 2019. Available online: https://blog.ml.cmu.edu/2019/11/12/federated-learning-challenges-methods-and-future-directions/ (accessed on 13 November 2020).
24. Mothukuri, V.; Parizi, R.M.; Pouriyeh, S.; Huang, Y.; Dehghantanha, A.; Srivastava, G. A survey on security and privacy of federated learning. *Future Gener. Comput. Syst.* **2021**, *115*, 619–640. [CrossRef]
25. Canetti, R.; Feige, U.; Goldreich, O.; Naor, M. Adaptively secure multi-party computation. In Proceedings of the Twenty-Eighth Annual ACM Symposium on Theory of Computing, Philadelphia, PA, USA, 22–24 May 1996; pp. 639–648.
26. Dwork, C. Differential privacy: A survey of results. In Proceedings of the International Conference on Theory and Applications of Models of Computation, Xi'an, China, 25–29 April 2008; Springer: Berlin/Heidelberg, Germany, pp. 1–19.
27. Tramèr, F.; Kurakin, A.; Papernot, N.; Goodfellow, I.; Boneh, D.; McDaniel, P. Ensemble adversarial training: Attacks and defenses. *arXiv* **2017**, arXiv:1705.07204.
28. Kairouz, P.; McMahan, H.B.; Avent, B.; Bellet, A.; Bennis, M.; Bhagoji, A.N.; Bonawitz, K.; Charles, Z.; Cormode, G.; Cummings, R.; et al. Advances and open problems in federated learning. *arXiv* **2019**, arXiv:1912.04977.
29. Yang, Q.; Liu, Y.; Chen, T.; Tong, Y. Federated machine learning: Concept and applications. *Acm Trans. Intell. Syst. Technol. (TIST)* **2019**, *10*, 1–19. [CrossRef]
30. Lim, W.Y.B.; Luong, N.C.; Hoang, D.T.; Jiao, Y.; Liang, Y.C.; Yang, Q.; Niyato, D.; Miao, C. Federated learning in mobile edge networks: A comprehensive survey. *IEEE Commun. Surv. Tutorials* **2020**, *22*, 2031–2063. [CrossRef]
31. Połap, D.; Srivastava, G.; Lin, J.C.W.; Woźniak, M. Federated Learning Model with Augmentation and Samples Exchange Mechanism. In Proceedings of the International Conference on Artificial Intelligence and Soft Computing, Zakopane, Poland, 20–24 June 2021; Springer: Berlin/Heidelberg, Germany, 2021; pp. 214–223.
32. Połap, D.; Srivastava, G.; Yu, K. Agent architecture of an intelligent medical system based on federated learning and blockchain technology. *J. Inf. Secur. Appl.* **2021**, *58*, 102748. [CrossRef]
33. Ahmed, U.; Srivastava, G.; Lin, J.C.W. Reliable customer analysis using federated learning and exploring deep-attention edge intelligence. *Future Gener. Comput. Syst.* **2022**, *127*, 70–79. [CrossRef]
34. Xu, R.; Baracaldo, N.; Zhou, Y.; Anwar, A.; Ludwig, H. Hybridalpha: An efficient approach for privacy-preserving federated learning. In Proceedings of the 12th ACM Workshop on Artificial Intelligence and Security, London, UK, 15 November 2019; pp. 13–23.
35. Kang, J.; Xiong, Z.; Niyato, D.; Zou, Y.; Zhang, Y.; Guizani, M. Reliable federated learning for mobile networks. *IEEE Wirel. Commun.* **2020**, *27*, 72–80. [CrossRef]
36. Zhu, H.; Zhang, H.; Jin, Y. From federated learning to federated neural architecture search: A survey. *Complex Intell. Syst.* **2021**, *7*, 639–657. [CrossRef]
37. Tian, Z.; Zhang, R.; Hou, X.; Liu, J.; Ren, K. FederBoost: Private Federated Learning for GBDT. *arXiv* **2020**, arXiv:2011.02796.
38. Chen, Y.; Qin, X.; Wang, J.; Yu, C.; Gao, W. Fedhealth: A federated transfer learning framework for wearable healthcare. *IEEE Intell. Syst.* **2020**, *35*, 83–93. [CrossRef]

39. Xu, J.; Glicksberg, B.S.; Su, C.; Walker, P.; Bian, J.; Wang, F. Federated learning for healthcare informatics. *J. Healthc. Inform. Res.* **2021**, *5*, 1–19. [CrossRef]
40. Yin, L.; Feng, J.; Xun, H.; Sun, Z.; Cheng, X. A Privacy-Preserving Federated Learning for Multiparty Data Sharing in Social IoTs. *IEEE Trans. Netw. Sci. Eng.* **2021**, *8*, 2706–2718. [CrossRef]
41. Li, L.; Fan, Y.; Tse, M.; Lin, K.Y. A review of applications in federated learning. *Comput. Ind. Eng.* **2020**, *149*, 106854. [CrossRef]
42. Brisimi, T.S.; Chen, R.; Mela, T.; Olshevsky, A.; Paschalidis, I.C.; Shi, W. Federated learning of predictive models from federated electronic health records. *Int. J. Med. Inform.* **2018**, *112*, 59–67. [CrossRef]
43. Nasajpour, M.; Karakaya, M.; Pouriyeh, S.; Parizi, R.M. Federated Transfer Learning For Diabetic Retinopathy Detection Using CNN Architectures. In Proceedings of the SoutheastCon 2022, IEEE, Mobile, AL, USA, 26 March–3 April 2022; pp. 655–660.
44. Li, Q.; Wen, Z.; Wu, Z.; Hu, S.; Wang, N.; Li, Y.; Liu, X.; He, B. A survey on federated learning systems: Vision, hype and reality for data privacy and protection. *IEEE Trans. Knowl. Data Eng.* 2021 , *early access*. [CrossRef]
45. Pfitzner, B.; Steckhan, N.; Arnrich, B. Federated Learning in a Medical Context: A Systematic Literature Review. *Acm Trans. Internet Technol. (TOIT)* **2021**, *21*, 1–31. [CrossRef]
46. Lyu, L.; Yu, H.; Yang, Q. Threats to federated learning: A survey. *arXiv* **2020**, arXiv:2003.02133.
47. Rahman, K.J.; Ahmed, F.; Akhter, N.; Hasan, M.; Amin, R.; Aziz, K.E.; Islam, A.M.; Mukta, M.S.H.; Islam, A.N. Challenges, applications and design aspects of federated learning: A survey. *IEEE Access* **2021**, *9*, 124682–124700. [CrossRef]
48. Li, T.; Sahu, A.K.; Talwalkar, A.; Smith, V. Federated learning: Challenges, methods, and future directions. *IEEE Signal Process. Mag.* **2020**, *37*, 50–60. [CrossRef]
49. Konečnỳ, J.; McMahan, H.B.; Yu, F.X.; Richtárik, P.; Suresh, A.T.; Bacon, D. Federated learning: Strategies for improving communication efficiency. *arXiv* **2016**, arXiv:1610.05492.
50. Zheng, S.; Shen, C.; Chen, X. Design and Analysis of Uplink and Downlink Communications for Federated Learning. *IEEE J. Sel. Areas Commun.* **2020**, *39*, 2150–2167. [CrossRef]
51. Luping, W.; Wei, W.; Bo, L. CMFL: Mitigating communication overhead for federated learning. In Proceedings of the 2019 IEEE 39th International Conference on Distributed Computing Systems (ICDCS), Dallas, TX, USA, 7–10 July 2019; pp. 954–964.
52. Chen, Y.; Sun, X.; Jin, Y. Communication-efficient federated deep learning with layerwise asynchronous model update and temporally weighted aggregation. *IEEE Trans. Neural Netw. Learn. Syst.* **2019**, *31*, 4229–4238. [CrossRef] [PubMed]
53. Shi, Y.; Yang, K.; Jiang, T.; Zhang, J.; Letaief, K.B. Communication-efficient edge AI: Algorithms and systems. *IEEE Commun. Surv. Tutorials* **2020**, *22*, 2167–2191. [CrossRef]
54. Li, T.; Sahu, A.K.; Zaheer, M.; Sanjabi, M.; Talwalkar, A.; Smith, V. Federated optimization in heterogeneous networks. *arXiv* **2018**, arXiv:1812.06127.
55. Sattler, F.; Wiedemann, S.; Müller, K.R.; Samek, W. Robust and communication-efficient federated learning from non-iid data. *IEEE Trans. Neural Netw. Learn. Syst.* **2019**, *31*, 3400–3413. [CrossRef]
56. Zhang, S.; Choromanska, A.E.; LeCun, Y. Deep learning with elastic averaging SGD. In Proceedings of the Advances in Neural Information Processing Systems, Montreal, QC, Canada, 7–12 December 2015; pp. 685–693.
57. Zhao, Y.; Li, M.; Lai, L.; Suda, N.; Civin, D.; Chandra, V. Federated learning with non-iid data. *arXiv* **2018**, arXiv:1806.00582.
58. Smith, V.; Forte, S.; Ma, C.; Takáč, M.; Jordan, M.I.; Jaggi, M. CoCoA: A general framework for communication-efficient distributed optimization. *J. Mach. Learn. Res.* **2017**, *18*, 8590–8638.
59. Briggs, C.; Fan, Z.; Andras, P. Federated learning with hierarchical clustering of local updates to improve training on non-IID data. *arXiv* **2020**, arXiv:2004.11791.
60. Singh, A.; Yadav, A.; Rana, A. K-means with Three different Distance Metrics. *Int. J. Comput. Appl.* **2013**, *67* , 13–17. [CrossRef]
61. Reisizadeh, A.; Mokhtari, A.; Hassani, H.; Jadbabaie, A.; Pedarsani, R. Fedpaq: A communication-efficient federated learning method with periodic averaging and quantization. In Proceedings of the International Conference on Artificial Intelligence and Statistics, PMLR, Online, 26–28 August 2020; pp. 2021–2031.
62. Karimireddy, S.P.; Kale, S.; Mohri, M.; Reddi, S.; Stich, S.; Suresh, A.T. Scaffold: Stochastic controlled averaging for federated learning. In Proceedings of the International Conference on Machine Learning, PMLR, Virtual, 13–18 July 2020; pp. 5132–5143.
63. Shamir, O.; Srebro, N.; Zhang, T. Communication-efficient distributed optimization using an approximate newton-type method. In Proceedings of the International Conference on Machine Learning, PMLR, Bejing, China, 22–24 June 2014; pp. 1000–1008.
64. Reddi, S.J.; Konečnỳ, J.; Richtárik, P.; Póczós, B.; Smola, A. AIDE: Fast and communication efficient distributed optimization. *arXiv* **2016**, arXiv:1608.06879.
65. Li, T.; Sahu, A.K.; Zaheer, M.; Sanjabi, M.; Talwalkar, A.; Smithy, V. Feddane: A federated newton-type method. In Proceedings of the 2019 IEEE 53rd Asilomar Conference on Signals, Systems, and Computers, Pacific Grove, California, USA, 3–6 November 2019; pp. 1227–1231.
66. AbdulRahman, S.; Tout, H.; Mourad, A.; Talhi, C. FedMCCS: Multicriteria client selection model for optimal IoT federated learning. *IEEE Internet Things J.* **2020**, *8*, 4723–4735. [CrossRef]
67. Xu, J.; Wang, H. Client selection and bandwidth allocation in wireless federated learning networks: A long-term perspective. *IEEE Trans. Wirel. Commun.* **2020**, *20*, 1188–1200. [CrossRef]
68. Nishio, T.; Yonetani, R. Client selection for federated learning with heterogeneous resources in mobile edge. In Proceedings of the ICC 2019-2019 IEEE International Conference on Communications (ICC), Shanghai, China, 20–24 May 2019; pp. 1–7.

69. Anh, T.T.; Luong, N.C.; Niyato, D.; Kim, D.I.; Wang, L.C. Efficient training management for mobile crowd-machine learning: A deep reinforcement learning approach. *IEEE Wirel. Commun. Lett.* **2019**, *8*, 1345–1348. [CrossRef]
70. Cho, Y.J.; Wang, J.; Joshi, G. Client Selection in Federated Learning: Convergence Analysis and Power-of-Choice Selection Strategies. *arXiv* **2020**, arXiv:2010.01243.
71. Zhang, W.Z.; Elgendy, I.A.; Hammad, M.; Iliyasu, A.M.; Du, X.; Guizani, M.; Abd El-Latif, A.A. Secure and Optimized Load Balancing for Multitier IoT and Edge-Cloud Computing Systems. *IEEE Internet Things J.* **2020**, *8*, 8119–8132. [CrossRef]
72. Kamp, M.; Adilova, L.; Sicking, J.; Hüger, F.; Schlicht, P.; Wirtz, T.; Wrobel, S. Efficient decentralized deep learning by dynamic model averaging. In Proceedings of the Joint European Conference on Machine Learning and Knowledge Discovery in Databases, Dublin, Ireland, 10–14 September 2018; Springer: Berlin/Heidelberg, Germany, 2018; pp. 393–409.
73. Bui, T.D.; Nguyen, C.V.; Swaroop, S.; Turner, R.E. Partitioned variational inference: A unified framework encompassing federated and continual learning. *arXiv* **2018**, arXiv:1811.11206.
74. Smith, V.; Chiang, C.K.; Sanjabi, M.; Talwalkar, A. Federated multi-task learning. *arXiv* **2017**, arXiv:1705.10467.
75. Guha, N.; Talwalkar, A.; Smith, V. One-shot federated learning. *arXiv* **2019**, arXiv:1902.11175.
76. Nguyen, H.T.; Sehwag, V.; Hosseinalipour, S.; Brinton, C.G.; Chiang, M.; Poor, H.V. Fast-convergent federated learning. *IEEE J. Sel. Areas Commun.* **2020**, *39*, 201–218. [CrossRef]
77. Tang, H.; Gan, S.; Zhang, C.; Zhang, T.; Liu, J. Communication compression for decentralized training. *arXiv* **2018**, arXiv:1803.06443.
78. Reisizadeh, A.; Taheri, H.; Mokhtari, A.; Hassani, H.; Pedarsani, R. Robust and communication-efficient collaborative learning. *arXiv* **2019**, arXiv:1907.10595.
79. He, C.; Tan, C.; Tang, H.; Qiu, S.; Liu, J. Central server free federated learning over single-sided trust social networks. *arXiv* **2019**, arXiv:1910.04956.
80. Lalitha, A.; Kilinc, O.C.; Javidi, T.; Koushanfar, F. Peer-to-peer federated learning on graphs. *arXiv* **2019**, arXiv:1901.11173.
81. Roy, A.G.; Siddiqui, S.; Pölsterl, S.; Navab, N.; Wachinger, C. Braintorrent: A peer-to-peer environment for decentralized federated learning. *arXiv* **2019**, arXiv:1905.06731.
82. Rothchild, D.; Panda, A.; Ullah, E.; Ivkin, N.; Stoica, I.; Braverman, V.; Gonzalez, J.; Arora, R. Fetchsgd: Communication-efficient federated learning with sketching. In Proceedings of the International Conference on Machine Learning, PMLR, Virtual Event, 13–18 July 2020; pp. 8253–8265.
83. Spring, R.; Kyrillidis, A.; Mohan, V.; Shrivastava, A. Compressing gradient optimizers via count-sketches. *arXiv* **2019**, arXiv:1902.00179.
84. Li, S.; Qi, Q.; Wang, J.; Sun, H.; Li, Y.; Yu, F.R. GGS: General Gradient Sparsification for Federated Learning in Edge Computing. In Proceedings of the ICC 2020-2020 IEEE International Conference on Communications (ICC), Virtual, 7–11 June 2020; pp. 1–7.
85. Hu, R.; Gong, Y.; Guo, Y. Sparsified Privacy-Masking for Communication-Efficient and Privacy-Preserving Federated Learning. *arXiv* **2020**, arXiv:2008.01558.
86. Sattler, F.; Wiedemann, S.; Müller, K.R.; Samek, W. Sparse binary compression: Towards distributed deep learning with minimal communication. In Proceedings of the 2019 IEEE International Joint Conference on Neural Networks (IJCNN), Budapest, Hungary, 14–19 July 2019; pp. 1–8.
87. Nicholls, J. Quantization in Deep Learning. 2018. Available online: https://medium.com/@joel_34050/quantization-in-deep-learning-478417eab72b/ (accessed on 28 April 2021).
88. Jiang, P.; Agrawal, G. A linear speedup analysis of distributed deep learning with sparse and quantized communication. *Adv. Neural Inf. Process. Syst.* **2018**, *31*, 2525–2536.
89. Amiri, M.M.; Gunduz, D.; Kulkarni, S.R.; Poor, H.V. Federated learning with quantized global model updates. *arXiv* **2020**, arXiv:2006.10672.
90. Koloskova, A.; Lin, T.; Stich, S.U.; Jaggi, M. Decentralized deep learning with arbitrary communication compression. *arXiv* **2019**, arXiv:1907.09356.
91. Dai, X.; Yan, X.; Zhou, K.; Yang, H.; Ng, K.K.; Cheng, J.; Fan, Y. Hyper-sphere quantization: Communication-efficient sgd for federated learning. *arXiv* **2019**, arXiv:1911.04655.
92. Shlezinger, N.; Chen, M.; Eldar, Y.C.; Poor, H.V.; Cui, S. Federated learning with quantization constraints. In Proceedings of the ICASSP 2020-2020 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP), Barcelona, Spain, 4–8 May 2020; pp. 8851–8855.
93. Shlezinger, N.; Chen, M.; Eldar, Y.C.; Poor, H.V.; Cui, S. UVeQFed: Universal vector quantization for federated learning. *IEEE Trans. Signal Process.* **2020**, *69*, 500–514. [CrossRef]
94. Liu, L.; Zhang, J.; Song, S.; Letaief, K.B. Hierarchical Quantized Federated Learning: Convergence Analysis and System Design. *arXiv* **2021**, arXiv:2103.14272.