ACCEPTED FROM OPEN CALL

Federated Learning and Wireless Communications

Zhijin Qin, Geoffrey Ye Li, and Hao Ye

Abstract

Federated learning becomes increasingly attractive in the areas of wireless communications and machine learning due to its powerful learning ability and potential applications. In contrast to other machine learning techniques that require no communication resources, federated learning exploits communications between the central server and the distributed local clients to train and optimize a model. Therefore, how to efficiently assign limited communication resources to train a federated learning model becomes critical to performance optimization. On the other hand, federated learning, as a brand-new tool, can potentially enhance the intelligence of wireless networks. In this article, we provide a comprehensive overview of the relationship between federated learning and wireless communications, including basic principles of federated learning, efficient communications for training a federated learning model, and federated learning for intelligent wireless applications. We also identify some research challenges and directions at the end of this article.

Introduction

With the success of machine learning (ML), especially deep learning (DL), in various areas, researchers in the communications community have also applied DL to realize intelligent communication system recently. In particular, DL can improve signal processing performance of communications systems [1]. Traditionally, a communication system consists of several modules, such as coding and decoding, modulation and demodulation, channel estimation, and signal detection. In addition to applying DL to each individual communication module, we can exploit *expert knowledge* in the area of wireless communications accumulated in the past century to simplify the structure of the DL model and speed up its convergence, which is called model-driven DL for physical layer communications. Moreover, a deep neural network (DNN) can be used to represent one or more modules. The whole transmitter or receiver can even be represented by a DNN, which is emerging as end-to-end communications. Furthermore, the DL-based end-to-end structure can be extended to semantic communication systems [2], which extracts and transmits the semantic meaning of sources, which could significantly improve communication efficiency.

DL can also be applied in resource allocation in communication networks. Traditionally, resource allocation is formulated as an optimization problem and then solved by applying optimization tools. Usually, the optimization problems formulated for wireless resource allocation are non-deterministic polynomial-time (NP) hard and therefore quite complicated to solve, if not impossible, to obtain the optimal solutions. DL can reduce the complexity and improve the performance on solving the optimization problem. In particular, deep reinforcement learning (DRL) can be directly exploited in resource allocation, where the environment contains channel quality, interference level, and so on, the action space includes spectrum access, power allocation, spatial resources, and so on, and the reward could be composed of latency, data rate, and so on. Through DRL, a good policy for resource allocation can be obtained to maximize the designed reward.

Note that most works on ML/DL for communications are based on centralized learning. Recently, federated learning [3–5], as a specific category of ML, has been proposed to perform model training distributedly at multiple participating clients, each with a part of training data, and coordinated by a central server. By doing so, the computation is offloaded from the central server to local clients. Moreover, in federated learning, the participating local clients communicate with the central server only on the model parameters learned locally rather than the raw data, which preserves privacy in addition to significant reduction in communication overhead. Therefore, federated learning is desired in many privacy-sensitive applications, such as training an image classification model based on the photos stored at different mobile devices.

In brief, federated learning is enabled by communications between the local clients and the central server and can also be used as a tool to improve the performance of wireless systems. This article discusses the relationship between federated learning and wireless communications with particular focus on the communication issues in federated learning as well as its applications in wireless communications. With this article, we aim to provide the readers a clear picture of the interplay between federated learning and wireless communications.

Digital Object Identifier:
10.1109/MWC.011.2000501

Zhijin Qin is with Queen Mary University of London; Geoffrey Ye Li is with Imperial College London; Hao Ye is with Georgia Institute of Technology.

Federated Learning

Even if some related work had been done before, the name *federated learning* was first used in [3] in 2016 for collaborative learning in wireless networks, where communication resources, such as bandwidth and transmission power, are limited and the privacy of local clients needs to be preserved. As shown in Fig. 1, there are overall K clients participating the learning. We aim to train a global model at the central server with parameters $\mathbf{w}$ by the whole dataset $\mathcal{D} = \cup_{k=1}^{K}\mathcal{D}_k$, where $\mathcal{D}_k$ is the raw data stored at client k. A straightforward way is to allow distributed local clients to send their data to the central server and then the model is trained centrally. However, local clients may be unwilling to share the raw data with others in some privacy-sensitive applications even if they are willing to participate in collaborative model training. Moreover, sending raw data to the central server also consumes significant communication resources, especially when the data size is huge and the number of participating local clients is large. Federated learning can be adopted to address the above issues.

Basic Principles

Following the broad definition in [4], federated learning is a machine learning setting where multiple local clients collaborate in training a model under the coordination of a central server while the raw data is kept at the local clients and only the model parameters are communicated between local clients and central server.

As shown in Fig. 1, federated learning includes the following four steps:

- *Local update:* Each client updates the learning model locally and in parallel according to its raw data.
- *Weight upload:* Each local client sends its intermediate results, that is, the updated parameters of the trained model $\mathbf{w}_k(t)$, to the central server.
- *Global aggregation:* The central server calculates the average weights, $\mathbf{w}(t)$, based on parameters received from local clients.
- *Weight feedback:* The server broadcasts the updated parameters to each local client for the next iteration.

By eliminating the transmission of raw data, federated learning addresses the privacy issue, reduces the communication overhead, and offloads the computation from the central server to local clients. In the following, we introduce the details involved in the aforementioned four steps.

Loss Function: Loss function, $\mathcal{L}(\mathbf{w}, x)$, is used to measure the prediction error corresponding to element x in a dataset for a machine learning model with parameters $\mathbf{w}$. The empirical loss function, defined as

$$F(\mathbf{w}) = \frac{1}{|\mathcal{D}|}\sum_{x\in\mathcal{D}}(\mathcal{L}\mathbf{w}, x),$$

is the average of loss functions with respect to all elements in a dataset, which obviously depends on both the model parameters, $\mathbf{w}$, and the dataset. Therefore, for data $\mathcal{D}_k$ stored at local client k, the corresponding empirical loss function,

$$F_k(\mathbf{w}) = \frac{1}{|\mathcal{D}|}\sum_{x\in\mathcal{D}}(\mathcal{L}\mathbf{w}, x),$$

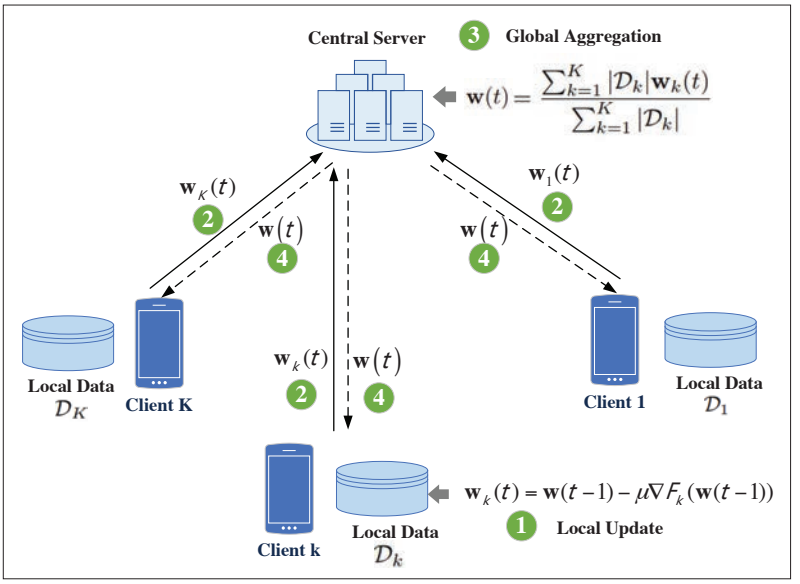

FIGURE 1. Basic principles of federated learning.

is usually different from those at other clients and the central server even for the learning model with the same parameters, $\mathbf{w}$. Then the global empirical loss function corresponding to the whole dataset, $\mathcal{D}$, can be expressed as:

$$F(\mathbf{w}) = \frac{\sum_{k=1}^{K}|\mathcal{D}_k|F_k(\mathbf{w})}{\sum_{k=1}^{K}|\mathcal{D}_k|}. \quad (1)$$

In the above discussion, we have used notations $|\mathcal{D}|$ and $|\mathcal{D}_k|$ for the numbers of elements in $\mathcal{D}$ and $\mathcal{D}_k$, respectively, and the fact that $|\mathcal{D}| = \cup_{k=1}^{K}|\mathcal{D}_k|$.

The learning process can be regarded as finding the model parameter, $\mathbf{w}$, to minimize the empirical loss function, $F(\mathbf{w})$, corresponding to a given dataset. Gradient descent is a popular approach to find the optimal parameter set, $\mathbf{w}^o = \arg\min F(\mathbf{w})$, and we will use it as an example in the subsequent discussion.

Model Weights: If using the gradient descent approach to minimize the global empirical loss function, the parameters of local models are averaged element-wise with weights proportional to sizes of the client datasets, which is given by:

$$\mathbf{w}(t) \coloneqq \mathbf{w}(t-1) - \mu\nabla F\big(\mathbf{w}(t-1)\big) = \frac{\sum_{k=1}^{K}|\mathcal{D}_k|\mathbf{w}_k(t)}{\sum_{k=1}^{K}|\mathcal{D}_k|}, \quad (2)$$

where μ is usually a small step size, $\nabla F(*)$ represents the gradient of $F(*)$, $\mathbf{w}(t)$ is the globally aggregated parameter set at the central server at time t, $\mathbf{w}_k(t)$ is the local parameter set of client k at the time slot t, which can be expressed as

$$\mathbf{w}_k(t) = \mathbf{w}(t-1) - \mu\nabla F_k(\mathbf{w}(t-1)). \quad (3)$$

Afterwards, the central server can calculate $\mathbf{w}(t)$ as long as the local gradient, $\nabla F_k(\mathbf{w}(t-1))$, is obtained. Therefore, only local gradients should be sent to the central server, which saves communication resources significantly, especially when gradient compression is used, as we will discuss later. However, we still have to feed the locally updated parameter set, $\mathbf{w}_k(t)$, to the central

In federated learning, the central server and local clients continue to exchange the updated model parameter sets during the training process, which consumes significant amounts of communication resources, especially when the distributed local clients are wireless connected devices and with a huge number, such as in IoT applications. To address the issue, we can compress the information to be transmitted and/or allocate limited communication resources efficiently.

server if the relationship between the updated local parameter set, $\mathbf{w}_k(t)$, and the previous global parameter set, $\mathbf{w}_k(t-1)$, is nonlinear. In this case, parameter aggregation becomes a more complicated function of $\mathbf{w}_k(t)$ for $k = 1, \cdots, K$ other than that in Eq. 2.

Federated Learning vs Distributed Learning

Depending on applications and settings, federated learning has different variants, such as cross-device and cross-silo federated learning [4]. Even if not unanimously agreed, some researchers regard federated learning as a kind of distributed learning. Nevertheless, the following prominent characteristics distinguish federated learning from the traditional distributed learning:

- Data is heterogeneous, that is, non independent and identically distributed (IID), and remains decentralized at different local clients.
- The clients are heterogeneous with various capacities.
- The central server conducts model training, but can never access the raw data at local clients due to the privacy issue.
- The central server has no control over the local clients and each client can decide whether to participate in the collaboration or not.

In contrast to federated learning, communication capability in traditional distributed learning is not a bottleneck since the central server and the local clients are usually connected to each other using wired lines or optical fibers, rather than through wireless channels. Most traditional distributed learning approaches were developed in order to offload computation to the local clients, rather than addressing the privacy or communications issues. Therefore, the whole dataset may be stored at the central server, and there could be raw data exchange between the central server and the local clients or among the local clients. The whole dataset can even be re-partitioned if required. In terms of the distribution scales, there are usually up to 1,000 clients in distributed learning while it could have as large as 10^{10} clients for federated learning [4].

To the best of our knowledge, there are no unanimous definitions of distributed learning, decentralized learning, or collaborative learning.

Practical Issues

In the following, we identify some issues that cannot be neglected in the design and applications of federated learning.

Stochastic Gradient Descent: Since loss function $F_k(\mathbf{w}(t))$ is determined by the raw data at the local client k, there is no closed-form expression usually. Therefore, it is impossible to analytically find the gradients, $\nabla F_k(\mathbf{w}(t))$. In other words, we cannot carry out one step update as in Eq. 3. Usually, stochastic gradient descent (SGD) is used to find the gradient iteratively at the local clients for each iteration between the central server and local clients. Readers can refer to [3, 6] for more details.

Robust Aggregation: For the parameter aggregation in Eq. 2, weighted average is used, which only considers the sizes of the raw data and is optimal only if the central server receives local parameter sets *accurately* and *simultaneously*. In reality, it is difficult to transmit parameter sets in full-precision. Nevertheless, there also exist corruption and noise during parameter exchanges between the central server and the local clients. Robust aggregation in [7] can address corrupted local parameter sets. As a result of heterogeneous computational capabilities and communication links for different local clients, there may be some local clients, called stragglers, that deliver their updated local parameter sets later than others and affect timely model parameter updates. The coded federated learning method [8] may partially deal with *stragglers* and speed up the convergence of model training.

Upload Frequency: In the above discussion, we assume that all local clients participate in parameter update at every iteration, which is somehow inefficient. For example, if the corresponding local parameters update is negligible, then its impact on the aggregation can be ignored. In wireless edge learning, even if the local parameters are non-trivial, the corresponding channel condition could be quite poor. As a result, large amounts of wireless resources, for example, transmission power and bandwidth, are required to compensate for channel distortion. In this situation, we should jointly consider the occupied communication resource and its contribution to the performance improvement to optimize the whole learning process [9].

In brief, federated learning is limited by the wireless links between the central server and local clients. Therefore, it is important to efficiently exploit limited wireless resources [10, 11], which will be discussed later. On the other hand, even if originally proposed to address the concerns related to privacy, device computation and storage, and communication bandwidth, federated learning has been applied in some wireless applications [2, 12, 13], such as wireless resource allocation, semantic communications, and localization, as detailed below.

Communication Issues in Federated Learning

In federated learning, the central server and local clients continue to exchange the updated model parameter sets during the training process, which consumes significant amounts of communication resources, especially when the distributed local clients are wireless connected devices and with a huge number, such as in Internet-of-Things (IoT) applications. To address the issue, we can compress the information to be transmitted and/or allocate limited communication resources efficiently. Many data compression and communication resource management techniques developed for general purposes can be used in federated learning. Furthermore, if the learning model is a neural network, we can perform neural network pruning and parameter pruning to further lower the communication costs during training. In this section, we introduce gradient compression, over-the-air computation, compression and aggregation joint design, and mobile edge computing, which fits federated learning well.

Gradient Compression

It has been reported in [6] that redundancy in the stochastic gradient could be as large as 99 percent in certain situations. Therefore, spectrum bandwidth can be significantly saved if the stochastic gradient is compressed properly. In addition to those general data compression techniques, especially compressive sensing, to address

the issue, gradient compression in [6] exploits the sparsity of stochastic gradient and has been designed specifically for federated learning.

Gradient compression includes quantization and sparsification. Gradient quantization converts gradient elements, which are usually continuous, into (low-precision) discrete values to facilitate digital transmission. A simple way is to quantify the original stochastic gradient into binary. Improved versions include three-level and four-level quantization. Gradient sparsification refers to removing the gradient elements with small amplitudes. An intuitive way is to turn the elements with amplitudes below a threshold into zero. However, it is sometimes hard to choose the threshold. Another way is to turn off a certain number of elements with small amplitudes into zero. It has been demonstrated that, by properly combining gradient quantization and sparsification, the compression ratio can reach as small as 2.5 percent in certain situations while reasonable convergence performance of model training is still maintained.

Deep gradient compression (DGC) proposed in [6] further improves the performance of compression by the following four steps: momentum correction, local gradient clipping, momentum factor masking, and warm-up training. DGC can achieve a compression radio of 0.17 percent in certain situations.

Note that the gradient compression performance strongly depends on the learning models. Most models currently studied are for image recognition and language processing. Therefore, it is not clear whether these results still hold for wireless applications.

Over-the-Air Computation

If multiple local clients transmit their updated local parameter sets through the same wireless channel simultaneously, the received signal at the central server will be the superposition of all the local parameter sets. Surprisingly, the addition or weighted average is computed over the air. On the other hand, the aggregated global parameter set depends only on the weighted average of the local parameter sets as in Eq. 2. Inspired by this observation, a group of works exploit the *over-the-air computation* property of the wireless multiple-access channel to obtain the weighted average, that is, to perform aggregation, directly at the central server, without requiring the individual local parameter set. By doing so, significant communication bandwidth can be saved, especially when the number of local clients is massive, as in IoT applications.

If there are multiple antennas at the receiver of the central server, there will be freedom to assign spatial resources for perfect parameter aggregation according to Eq. 2. Figure 2 demonstrates the principle of the method in [9], where the receiver at the central server is with an antenna array while the transmitter at each local client is with one antenna. In the *i*-th time slot, the local clients simultaneously send their *i*-th elements of the local parameters, with proper power scales, through wireless channels. Then the received signal vector in the *i*-th time slot at the central server will be the superimposition of the signals from different clients. By properly selecting the beamforming vector, the desired weighted summation for perfect aggregation for the *i*-th local parameter can be achieved.

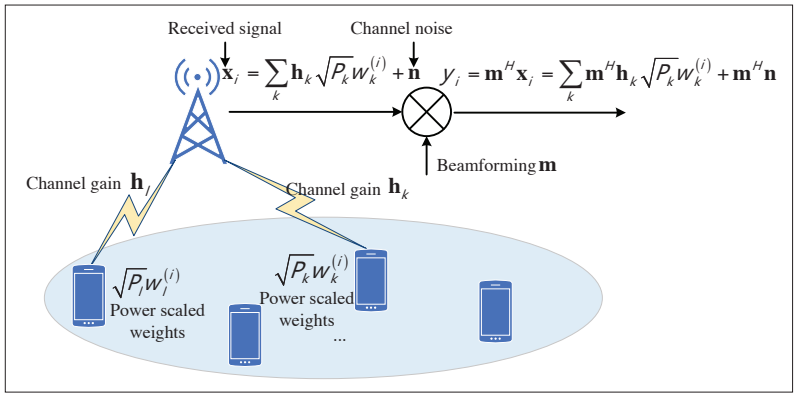

FIGURE 2. Parameter aggregation through over-the-air computation and exploiting spatial freedom.

With channel distortion and a maximum transmission power constraint for each local client in reality, there may not be enough freedom for perfect aggregation, especially when there are a huge number of local clients. In [9], a sparse and low-rank optimization problem is formulated to address the issue. The effective approach in [9] selects a subset of local clients with proper transmission power and carefully adjusts the beamforming vector to optimize the statistical performance of federated learning. If multiple antennas are equipped in both the clients and the central server, the transmitting and receiving beamforming matrices optimization to minimize the MSE at the receiver is NP-hard in general. Practical close-to-optimal solutions have been developed to address this issue [14].

The over-the-air computation has been applied for various machine learning algorithms, where the input is a group of samples and the order of the samples does not affect the output [15].

Joint Local Compression and Global Aggregation

In [10], all local clients compress their gradients by compressive sensing and then send to the central server through the same wireless channel simultaneously. The central server then reconstructs and aggregates the gradient information from the noisy observation, which combines gradient compression and over-the-air computation. Figure 3 compares the communication gain of the over-the-air computation approach (OverAir) and the CSOverAir, where the communication gain is measured as the ratio of the communication overhead of a conventional error-free communication system[1] and that of the OverAir and CSOverAir. With OverAir, all the devices communicate with the server simultaneously using over-the-air computation. Therefore, the communication cost for each training iteration can be reduced with a factor of the number of devices. 3 dB and 10 dB are used to show performance gain for over-the-air computation under low and high SNR areas, which are chosen based on the widely used SNR ranges in the existing works [9, 14]. Although the channel noise slows down the convergence, the overall communication overhead has been reduced significantly and the gain increases as the number of devices grows, as shown in Fig. 3. In CSOverAir, the gradients on each device are first sparsified and only 1 percent of the gradients are kept. Compressive

[1] In the conventional error-free communication system, the devices send their own gradients separately to the server in each iteration. The gradients are represented in float32 and the communication cost for error-free transmission of each device is lower bounded by $B = 32 \times M/C$, where M is the number of parameters and C is the channel capacity, $C = \log_2(1 + \text{SNR})$.

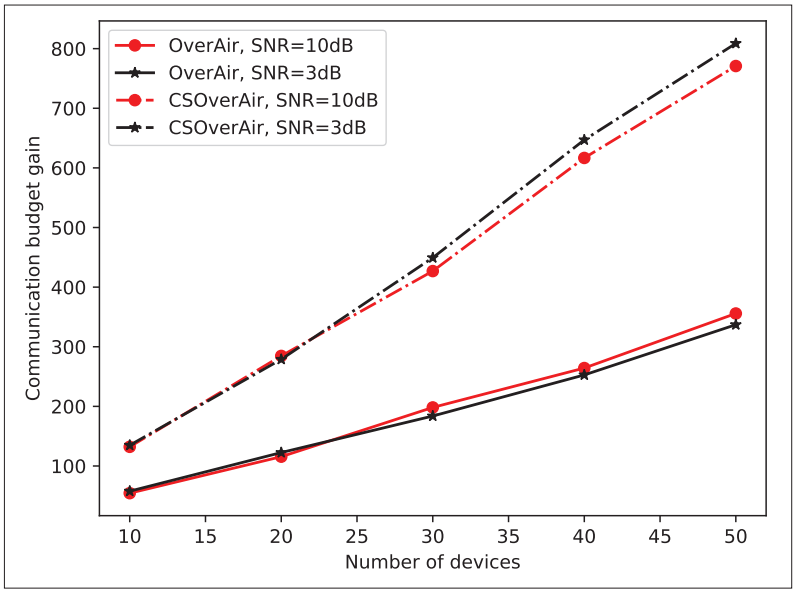

FIGURE 3. Communication budget gain of the over-the-air computation with parameter compression at the local clients and recovery/aggregation at the central server.

sensing is then used for compressing the sparsified gradients to save the communication cost. All the devices send the compressed gradients to the server simultaneously using over-the-air computation and the sum of the sparsified gradients is recovered at the center server. From the figure, the communication cost can be further reduced significantly with CSOverAir.

The frequency of global parameter aggregation is linked to the communication overhead and the convergence speed of model training. In [11], the convergence of gradient-descent based federated learning is analyzed and a convergence bound is obtained. Then, a control algorithm is developed based on the convergence bound. After learning data distribution, system dynamics, and model characteristics, the control algorithm adjusts the frequency of aggregation to minimize the global loss function with given communication resources. To efficiently exploit limited communication resources, joint optimization of parameter compression, communication resource allocation, and model training/aggregation can be performed, which is also a promising direction for further research.

Mobile Edge Computing

In federated learning, participating clients are heterogeneous with various data sizes and qualities, computation capacities, and willingness to participate. Communication cost is one of the bottlenecks in federated learning. It is natural to bring mobile edge computing into federated learning to leverage the communication and computation costs. With mobile edge computing in federated learning, edge servers are introduced for parameter aggregation from a group of clients locally, given that the latency from clients to the edge server is lower than that of client-cloud links. Considering the heterogeneity of clients, various resources should be carefully allocated to improve the training efficiency, such as client grouping, joint radio and computation resource management, and adaptive aggregation.

Federated Learning for Wireless Applications

Federated learning has been applied to various wireless scenarios. In this section, we will provide three examples: vehicular communications [12], localization [13], and semantic communications [2].

Distribution Estimation for Vehicular Communications

Vehicular communications will make our daily vehicular operation safer, greener, and more efficient, which also paves the path to intelligent driving. In addition to communicating with the infrastructure (V2X), vehicles also need to exchange critical information, such as safety related messages, with their surrounding vehicles (V2V) with ultra reliability and low latency, which is referred to as ultra-reliable low-latency communication (URLLC).

To ensure URLLC by optimally allocating limited communication resources, that is, frequency bands and transmission power, we have to model and capture events with extremely low probability, such as the probability of queue length over a threshold, and find their relationship with the assigned communication resources. Considering the high dynamics and high mobility of vehicular networks, it is often impossible to find an exact or closed-form expression. Fortunately, extreme value theory (EVT) can address the issue [12], which models a rare event by the generalized Pareto distribution determined by several critical parameters and converts the issue into estimating these critical parameters.

Federated learning has been used in [12] to estimate these critical parameters and catch the relationship between assigned resources, link reliability, and transmission latency. Different from the classic maximum likelihood estimation (MLE) that transmits data from different vehicular users to roadside units, the critical parameters can be obtained using federated learning based MLE, which can save communication resources for data transmission and avoid privacy issues. It has been demonstrated that federated learning based estimation can save up to 79 percent overhead but with estimation accuracy similar to the central solution.

Distributed Localization

For a given environment, radio features of a mobile device can uniquely determine its location, which makes it possible to perform localization based on radio features. However, the relationship between the mobile location and the corresponding radio features is usually very complicated. A deep learning model can be used to map the radio features into a specific location. To train the deep learning model, a huge training data on radio features corresponding to different locations is required, which is sometimes challenging. One way is to collect the data on radio features and mobile locations from all mobiles in a certain area and then use them to train the deep learning model for localization, which, however, will cause privacy issues and huge communication overhead. Therefore, federated learning is used in [13] to train machine learning models for localization, which is called federated localization (FEDLOC).

The framework of FEDLOC is similar to generic federated learning in Fig. 1. Each mobile, as a local client, collects local data on radio features and

Pruned Model	BLEU score with $m = 4$	ψ	BLEU score with $m = 8$	ψ	BLEU score with $m = 16$	ψ	BLEU score with $m = 32$	ψ
$\gamma = 0$	0.811194	8	0.906763	4	0.903089	2	0.895602	1
$\gamma = 0.6$	0.835863	20.0	0.897143	10.0	0.900468	5.0	0.9093	2.5
$\gamma = 0.9$	0.810322	80.0	0.895306	40.0	0.910554	20.0	0.89515	10

TABLE 1. The BLEU score and compression ratio, ψ, comparisons versus different sparsity ratio, γ, and quantization level, m, SNR = 12dB [2].

locations, updates the model parameter set locally, and sends it to the central server. The base station or just a fusion center, as the central server, aggregates the received local parameter sets to obtain the global one. After comparing the two machine learning models for localization, it is found in [13] that the *Gaussian processes* model with maximum likelihood loss function is better than the DNN model with least-square loss function based on the testing by real data.

Distributed Semantic Communication Systems

Semantic communication, as a revolution of conventional communication, interprets information at the semantic level. Instead of aiming at recovering data accurately in conventional communications, semantic communications are application-oriented by designing communication and application jointly with particular attention to the meanings of source messages, where only useful, relevant, and important information with respect to the applications is transmitted.

A deep learning enabled semantic communication (DeepSC) system [2] is designed for text transmission, where the meaning of text is extracted and compressed at the transmitter and the receiver recovers the meaning of the text rather than the text itself. DeepSC has shown huge potential for text transmission, especially at the low signal-to-noise (SNR) region. By implementing the DeepSC in a distributed way for IoT networks, the cloud/edge platform performs DeepSC model training and updating while IoT devices perform data collection and transmission based on the trained model. To make it affordable for IoT devices, a lite distributed semantic communication system, named L-DeepSC, has been proposed for text transmission with low complexity. As shown in Fig. 4, L-DeepSC consists of the following three steps:

- The cloud/edge platform trains an initial model or updates the trained model by received semantic features.
- The trained/updated model is broadcast to all participating IoT devices.
- Each participating device uploads the semantic features based on the trained model and its raw data.

In particular, by pruning the model redundancy and lowering the weight resolution, L-DeepSC is affordable for power-constrained IoT devices, and the bandwidth required for model weight transmission between local IoT devices and the cloud/edge is reduced significantly.

Simulation results demonstrate that the L-DeepSC could achieve as high as 40× compression ratio without performance degradation. As shown in Table 1, the BLEU score is a metric for evaluating the received sentence to a reference sentence,

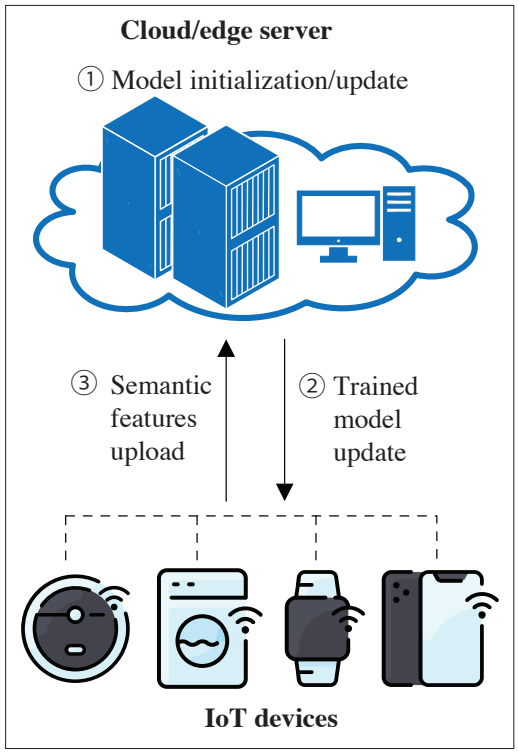

FIGURE 4. The framework of distributed semantic communication system for IoT networks.

that is, the transmitted one. For quantization level $m = 8$, sparsity ratio $\gamma = 0.9$, the compression ratio $\psi = 40$, the size of model parameters is reduced significantly while the BLEU score is similar as that in the case with $m = 32$ and $\psi = 1$. As a result, the communication cost between IoT devices and the center is reduced significantly. Furthermore, the privacy between IoT devices and the cloud/edge platform can be guaranteed for free as only the model weights and semantic features of texts are transmitted.

From the above three examples, the power of federated learning in wireless applications has been demonstrated in terms of saving communication costs, protecting data privacy, and extending the lifetime of devices. For different wireless applications, it could provide further benefits based on specific design. Nevertheless, the applications of federated learning are still in its infancy, which deserves more investigations.

Conclusion and Remarks

In this article, we have presented the principle and efficient communications of federated learning. We have also discussed some wireless application examples, which reflect the power of federated learning for future wireless systems. However,

Instead of aiming at recovering data accurately in conventional communications, semantic communications are application-oriented by designing communication and application jointly with particular attention to the meanings of source messages, where only useful, relevant, and important information with respect to the applications is transmitted.

The privacy for federated learning should be improved with extra efforts, such as employing classic cryptographic protocols for communications and applying differential privacy. However, these approaches often degrade the model performance or introduce additional communication costs. How to balance privacy, accuracy, and communication costs requires further investigation.

many issues and challenges remain unexplored. Here we identify some of them.

Over-the-Air Computation: It has been proposed for joint uplink communication and aggregation [9, 10], where perfect synchronization and accurate channel state information are assumed. In reality, exact synchronization is impossible and the channel state information used for receiver beamforming is usually contaminated by channel noise and distortion. Furthermore, the wireless channel may have multi-path delay spread. It is desired to address these deployment imperfections on over-the-air computation for model parameter aggregation.

Local Update and Global Aggregation: Instead of sharing all the weights with the server, each local client can keep its own model parameters, which are close to but not the same with the weights of the global model at the server. Models at local clients can be tailored for their own purposes. Therefore, a key problem is how to update the local models with the broadcast information from the server and how to process and transmit the gradients of the local model to help the training of the global model at the server. One possible solution is using a meta-learner for learning the gradients for local and global models.

Privacy: The privacy of federated learning has been improved significantly compared with sending the raw data to the central server straightforwardly, but there is still partial privacy leakage. Therefore, the privacy for federated learning should be improved with extra efforts, such as employing classic cryptographic protocols for communications and applying differential privacy. However, these approaches often degrade the model performance or introduce additional communication costs. How to balance privacy, accuracy, and communication costs requires further investigation.

New Wireless Applications: A wireless network can be regarded as a distributed learning system, which fits in well with applications of federated learning. In addition to distributed localization [13] and distributed semantic communications [2], more wireless applications are expected, such as mobile edge caching and resource allocation in vehicular networks.

References

[1] Z.-J. Qin *et al.*, "Deep Learning in Physical Layer Communications," *IEEE Wireless Commun.*, vol. 26, no. 2, Apr. 2019, pp. 93–99.

[2] H.-Q. Xie and Z.-J. Qin, "A Lite Distributed Semantic Communication System for Internet of Things," *IEEE JSAC*, vol. 39, no. 1, Jan. 2021, pp. 142–53.

[3] H. B. McMahan *et al.*, "Communication-Efficient Learning of Deep Networks from Decentralized Data," *Proc. PMLR*, 2017.

[4] P. Kairouz *et al.*, "Advances and Open Problem in Federated Learning," https://arxiv.org/abs/1912.04977v1, Dec. 2019.

[5] P. Bellavista, L. Foschini, and A. Mora, "Decentralised Learning in Federated Deployment Environments: A System-Level Survey," *ACM Computing Surveys*, vol. 54, no. 1, Feb. 2021.

[6] Y.-J. Lin *et al.*, "Deep Gradient Compression: Reducing the Communication Bandwidth for Distributed Training," *Proc. ICLR*, 2018.

[7] K. Pillutla, S. M. Kakade, and Z. Harchaoui, "Robust Aggregation for Federated Learning," https://arxiv.org/abs/1912.13445v1, Feb. 2020.

[8] M. R. Sprague *et al.*, "Asynchronous Federated Learning for Geospatial Applications," *Proc. Joint European Conf. Machine Learning and Knowledge Discovery in Databases*, Springer, Cham, 2018.

[9] K. Yang *et al.*, "Federated Learning via Over-the-Air Computation," *IEEE Trans. Wireless Commun.*, vol. 19, no. 3, Mar. 2020, pp. 2022–35.

[10] M. Mohammadi and D. Gündüz, "Machine Learning at the Wireless Edge: Distributed Stochastic Gradient Descent Over-the-Air," *IEEE Trans. Signal Process.*, vol. 68, Mar. 2020, pp. 2155–69.

[11] S.-Q. Wang *et al.*, "Adaptive Federated Learning in Resource Constrained Edge Computing Systems," *IEEE JSAC*, vol. 37, no. 6, June 2019, pp. 1205–21.

[12] S. Samarakoon *et al.*, "Distributed Federated Learning for Ultra-Reliable Low-Latency Vehicular Communications," *IEEE Trans. Commun.*, vol. 68, no. 2, Feb. 2020, pp. 1146–59.

[13] F. Yin *et al.*, "FEDLOC: Federated Learning Framework for Cooperative Localization and Location Data Processing," https://arxiv.org/abs/2003.03697v1, Mar. 2020.

[14] G. Zhu and K. Huang, "MIMO Over-the-Air Computation for High-mobility Multi-Modal Sensing," *IEEE Internet Things J.*, vol. 6, no. 4, Aug. 2019, pp. 6089–6103.

[15] H. Ye, G. Y. Li, and B. H. Juang "Deep Over-the-Air Computation," *Proc. IEEE Globecom*, Dec. 2020, pp. 1–6.

Biographies

Zhijin Qin is a lecturer at Queen Mary University of London. She was with Lancaster University as a lecturer and Imperial College London as a research associate from 2016 to 2018. Her research interests include semantic communications, end-to-end communications, compressive sensing, and intelligent resource allocation in emerging applications. She serves as an area editor for the IEEE JSAC Series on Machine Learning for Communications and Networks, an associate editor of *IEEE Transactions on Communications*, *IEEE Communications Letters*, and *IEEE Transactions on Cognitive Communications and Networking*. She received the IEEE GLOBECOM Best Paper Award in 2017 and IEEE SPS Young Author Best Paper Award in 2018.

Geoffrey Ye Li is a Chair Professor with Imperial College London. His general research is in signal processing and machine learning for wireless communications. In related areas, he has published over 600 articles with over 47,000 citations and been listed as a Highly-Cited Researcher by Thomson Reuters. He has been an IEEE Fellow since 2006. He won IEEE ComSoc S. O. Rice Prize Paper Award, the Award for Advances in Communication, and the Edwin Howard Armstrong Achievement Award, IEEE VTS James Evans Avant Garde Award and Jack Neubauer Memorial Award, IEEE SPS Donald G. Fink Overview Paper Award, and Distinguished ECE Faculty Achievement Award from Georgia Tech.

Hao Ye received the B.E. degree in information engineering from Southeast University, Nanjing, China, in 2012, the M.E. degree in electrical engineering from Tsinghua University, Beijing, China, in 2015, and the Ph.D. degree in electrical and computer engineering from the Georgia Institute of Technology, Atlanta, GA, USA, in 2020. Since 2021, he has been a researcher with Qualcomm AI Research, San Diego, CA, USA. His general research interests include wireless communications, signal processing, and machine learning.